UNCLASSIFIED

DTIC FILE COPY

SECURITY CLASSIFICATION OF THIS PAGE (When Data Entered)

AD-A196 414

REPORT DOCUMENTATION PAGE		READ INSTRUCTIONS BEFORE COMPLETING FORM
1. REPORT NUMBER AFIT/CI/NR 88-76	2. GOVT ACCESSION NO.	3. RECIPIENT'S CATALOG NUMBER
4. TITLE (and Subtitle) USE OF A PORTABLE MICROCOMPUTER AS A DATA COLLECTION TOOL TO SUPPORT INTEGRATED SIMULATION SUPPORT ENVIRONMENTS: A CONCEPT		5. TYPE OF REPORT & PERIOD COVERED MS THESIS
		6. PERFORMING ORG. REPORT NUMBER
7. AUTHOR(s) JAMES EDWARD WILKINSON		8. CONTRACT OR GRANT NUMBER(s)
9. PERFORMING ORGANIZATION NAME AND ADDRESS AFIT STUDENT AT: UNIVERSITY OF CENTRAL FLORIDA		10. PROGRAM ELEMENT, PROJECT, TASK AREA & WORK UNIT NUMBERS
11. CONTROLLING OFFICE NAME AND ADDRESS		12. REPORT DATE 1988
		13. NUMBER OF PAGES 111
14. MONITORING AGENCY NAME & ADDRESS (if different from Controlling Office) AFIT/NR Wright-Patterson AFB OH 45433-6583		15. SECURITY CLASS. (of this report) UNCLASSIFIED
		15a. DECLASSIFICATION/DOWNGRADING SCHEDULE

16. DISTRIBUTION STATEMENT (of this Report)

DISTRIBUTED UNLIMITED: APPROVED FOR PUBLIC RELEASE

DTIC ELECTE AUG 0 3 1988 S D H

17. DISTRIBUTION STATEMENT (of the abstract entered in Block 20, if different from Report)

SAME AS REPORT

18. SUPPLEMENTARY NOTES

Approved for Public Release: IAW AFR 190-1
LYNN E. WOLAVER Lynn Wolaver 19 July 88
Dean for Research and Professional Development
Air Force Institute of Technology
Wright-Patterson AFB OH 45433-6583

19. KEY WORDS (Continue on reverse side if necessary and identify by block number)

20. ABSTRACT (Continue on reverse side if necessary and identify by block number)

ATTACHED

DD FORM 1 JAN 73 1473 EDITION OF 1 NOV 65 IS OBSOLETE

UNCLASSIFIED

SECURITY CLASSIFICATION OF THIS PAGE (When Data Entered)

ABSTRACT

Advances in simulation methodology, and the computer support systems to implement these methodologies, have led to the development of integrated simulation support environments. These environments, or collection of software tools, seek to aid the analyst in developing models, data management and analysis, and data presentation.

While an integrated simulation environment provides a more organized structure for managing and performing simulation projects, and provides a database management structure for storing, manipulating, and analyzing data, they do not address the actual process of going out and obtaining the data. As a result, many of the common problems associated with poor problem and system definition, and low quality model input data, may still occur.

To solve this problem, this study examines the concept of developing a "support-support" system; a portable microcomputer with software tools designed to support collection of the data, both subjective and objective, required in a simulation study. This data can then be ported into the integrated support system for analysis and model development.

In developing this concept, the simulation process is better defined using structured analysis diagrams. Based on this analysis the functions that a support-support system

could best accomplish are identified and a conceptual specification developed. An implementation strategy is proposed which is based on the use of readily available software tools, such as dBASE III, and the use of a simple programming language, such as BASIC.

To demonstrate how this strategy can be implemented, a BASICA program was developed to support model input data collection. Using a graphic display to define input data requirements and single key inputs, this program should maximize the time an analyst can spend observing the system and minimize the time he/she has to spend entering data.

THE USE OF A PORTABLE MICROCOMPUTER AS A DATA COLLECTION TOOL TO SUPPORT INTEGRATED SIMULATION SUPPORT ENVIRONMENTS: A CONCEPT

BY

JAMES EDWARD WILKINSON
B.S., North Dakota State University, 1973

THESIS

Submitted in partial fulfillment of the requirements
for the Master of Science degree in Engineering
in the Graduate Studies Program
of the College of Engineering
University of Central Florida
Orlando, Florida

Spring Term
1987

ABSTRACT

Advances in simulation methodology, and the computer support systems to implement these methodologies, have led to the development of integrated simulation support environments. These environments, or collection of software tools, seek to aid the analyst in developing models, data management and analysis, and data presentation.

While an integrated simulation environment provides a more organized structure for managing and performing simulation projects, and provides a database management structure for storing, manipulating, and analyzing data, they do not address the actual process of going out and obtaining the data. As a result, many of the common problems associated with poor problem and system definition, and low quality model input data, may still occur.

To solve this problem, this study examines the concept of developing a "support-support" system; a portable microcomputer with software tools designed to support collection of the data, both subjective and objective, required in a simulation study. This data can then be ported into the integrated support system for analysis and model development.

In developing this concept, the simulation process is better defined using structured analysis diagrams. Based on this analysis the functions that a support-support system

could best accomplish are identified and a conceptual specification developed. An implementation strategy is proposed which is based on the use of readily available software tools, such as dBASE III, and the use of a simple programming language, such as BASIC.

To demonstrate how this strategy can be implemented, a BASICA program was developed to support model input data collection. Using a graphic display to define input data requirements and single key inputs, this program should maximize the time an analyst can spend observing the system and minimize the time he/she has to spend entering data.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

CHAPTER 1

INTRODUCTION

Background

In recent years there have been numerous advances in the "art and science" of simulation and associated software. New simulation languages and methodologies have been developed that reflect a "shift from the program to the model view of the simulation process" (Nance 1983). There has also been an increased interest in the development of support software and simulation "environments" that comprise integrated collections of these software tools (Henricksen 1984). One of the more visible areas has been in the area of interactive simulation involving graphics and animation.

In regards to simulation environments, an integrated environment is a collection of software tools for designing, writing, and validating models; writing and verifying data; and designing and carrying out experiments with models. An important aspect of an integrated environment is that the user is shielded from being a computer technician, i.e., having to spend a significant amount of time managing files, making sure data are in correct formats, etc. (Henricksen 1984; Standridge and Walker 1983).

Additionally termed as integrated simulation support systems, they seek to "deal with ... other aspects of

performing a simulation project" such as model building, user data management, data analysis, and data presentation (Standridge and Walker, 1983). A primary attribute is integrated data management, which seeks to address the needs of newer simulation methodologies and the complexity of the systems now being modeled (Standridge 1983; Evers Bachert, and Santucci 1981). Standridge (1985) states that the requirements for these systems have evolved to include: 1) a framework which provides a more standardized way for performing a simulation project; 2) a system which provides effective user interfaces; and 3) a system which is integrated.

Such a system, coupled with a effective methodology, can alleviate many of the problems and failures of simulation projects. These problems include the need to effectively communicate model formulation and assumptions; poor communication, both with user and project team members; "failure to use modern tools and techniques to manage the development of a large, complex computer program"; and failure to adequately describe the system (Evers, Bachert, and Santucci 1981; Annino and Russell 1979; Hahn and Comer 1981).

Other just as common problems result from poor or inadequate data, where data is defined in its broadest context to include subjective data, such as information about the problem, objectives, and system to be modeled; as well as objective data such as that used to define variable/para-

meter inputs and to validate the model. Some of these problems include:

1) A poor and incomplete definition of the problem.

2) Failure to identify the objectives of the model.

3) Failure to adequately identify the data needed to execute the model and how this data is to be collected.

4) Using statistical procedures that the data is independent and identically distributed when is is not.

5) Assuming observations in the data set are homogeneous when they are not.

(Miller and Pare 1986; Evers, et al. 1981; Shannon 1975).

Problem/Scope

While an integrated simulation environment provides a more organized structure for managing and performing simulation projects, and provides a database management structure for storing, manipulating, and analyzing data, they do not address the actual process of going out and obtaining the data. Thus many of the above mentioned problems may still occur.

This research effort has focused on better defining the simulation process, particularly the "upfront" process, with an eye towards developing a system specification for what could be termed a support-support system. The objective or purpose of such a system would be to provide a more organized or disciplined approach and tool for the data collection phases of a simulation project. This would in turn reduce the chances of failure due to poor or inadequate data.

Assumptions

The development of this topic and problem statement were motivated by and based on the following assumptions. They also have influenced the approach taken in trying to resolve this problem. While it is felt that these assumptions (except no. 10) are supported by other sources, they will be stated without justification at this point and addressed separately throughout this report. Some have already been stated or implied, but will be restated here for clarity.

1. Data collection also involves the collection of subjective data such as, understanding the user's problem, identifying system components and how they interact, identifying key decision makers, etc.

2. Data collection is a very important and often times underestimated part of the simulation process. It is also sometimes overlooked or assumed away.

3. An integrated simulation support system, such as the Extended Simulation System (TESS) is a valuable aid in properly executing a simulation project.

4. The new simulation analyst is very poorly prepared to properly undertake a simulation project. Most simulation texts, while acknowledging the importance of the initial phases of the simulation process do not expand on how to do it but rather focus on simulation as a software development process (programming exercise).

5. There is not an integrated support system that currently supports this collection of subjective data, nor objective data; unless one considers the more sophisticated data acquisition systems that automatically collect and archive system data.

6. Despite the existence, and ideally the benefits, of using a computer controlled data acquisition system, there are still circumstances that require collection by observation. This is particularly true when dealing with subjective data which involves observing, interviewing, and researching.

7. The microcomputer can be a powerful tool in carrying out simulation studies.

8. The potential of the "laptop" portable microcomputer has not been fully exploited. Furthermore, the portable microcomputer could form the basis for an integrated support-support system.

9. Experience and an effective methodology (procedures) are key elements in overcoming many of the failures of simulation projects. However, a system which incorporates the experience of experts and is based on proper procedure could overcome, in part, a lack of experience.

10. For computer software to be accepted and used by a prospective user it must not only be adaptable to to the task to be accomplished, but also to the particular individual user.

Approach

In approaching this problem it was felt the future is represented by integrated support systems and while data acquisition systems feeding the central data base of such a system is the ideal, this is not always possible. The next step down from this would be a portable computerized system which was designed along the same lines as the support system and could be used by the analyst to guide his data and information collection efforts. The results could then be ported to the main support system for final analysis and use in modeling the system and running the simulation. It is felt such a system can provide an organized and structured framework for the early phases of the project which in some cases would simplify the process and make it more efficient, and in all cases aid in avoiding some of the more common mistakes.

The study thus began with a look at the simulation process with emphasis on the early phases. The process was then further broken down so as to identify all the tasks that would have to be performed during the investigation and collection phase. Those tasks that could then be computerized were identified.

Next, the status and concepts of simulation support environments were investigated. Combining these concepts with the tasks to be performed, a system specification was developed which defined the requirements of a support-support system. This specification was also based on the current use of portable computers/data collectors, particularly in the area of work measurement.

Finally, because time prevented the building of an entire system, a generic data collection program was developed which was based on the concepts detailed in the system specification. Because the bulk of the information currently available deals with the collection of objective data (numbers) this software supports the collection of such data. This is not to imply that such a system cannot or should not support the collection of subjective data. It should and it must in order to be a complete and useful system.

Report Overview

This report will generally follow the approach outlined above. Chapters 2 discusses the problem in more depth. Chapter 3 looks at the simulation process and describes the procedure used to better define this process. Chapter 4 then

presents a generic system specification for a support-support system and Chapter 5 discusses how this specification could conceptually be implemented. Chapter 6 then takes a detailed look at an input collection program which partially implements the system specification. Because much remains to be done before a complete system exists, Chapter 7 discusses areas for future research.

CHAPTER 2

PROBLEM DISCUSSION

Introduction

This chapter will expand on the background material and problem statement provided in Chapter 1. It represents a brief review of the literature and also a more complete justification of the problem. The next section will begin with a personalized view of the problem and then expand this discussion to involve other problems that confront the simulation analyst. Then some of the methodologies that have been proposed to solve these problems, and the more sophisticated support systems that support these methodologies will be discussed. Finally, the last section will come full circle in that despite tremendous advances in both methodologies and support systems, the problem outlined in first section still essentially exists.

The Problem -- A Personal View

Simulation is "the process of designing a model of a real system and conducting experiments with this model for the purpose either of understanding the behavior of the system or of evaluating various strategies (within the limits imposed by a criterion or set of criteria) for the operation

of the system" (Shannon 1975). In discussing the simulation process most texts and articles (Banks and Carson 1984; Gordon 1978; Law and Kelton 1982; Schmidt 1985; Shannon 1975; and others) outline a series of steps "to guide a model builder in a thorough and sound simulation study" (Banks and Carson 1984). An example from Shannon (1975) is shown in Figure 1.

Depending on the emphasis of the book/article, each of the steps are discussed in varying levels of detail. Texts generally focus on the more technical aspects, such as programming and statistical analysis of input/output data. The more subjective aspects of the process, such as, problem definition, identifying objectives, system definition, and how to go about collecting this information and other data, often receive very little attention. This is meant to be more of an observation than a criticism, because it makes sense to spend the most time on concepts that are, for the most part, new to a student. It is felt, however, that this can result in a skewed perception of what a simulation study is, or why a study is undertaken.

Shannon (1975) and to a degree Martin (1968) do provide a more expanded discussion of some of these more subjective aspects. Shannon particularly focuses on simulation as a complete process, i.e., the study is not done until its results are accepted and implemented. He also discusses the political aspects of having the results of a study accepted by the decision makers. Shannon (1975) does acknowledge that

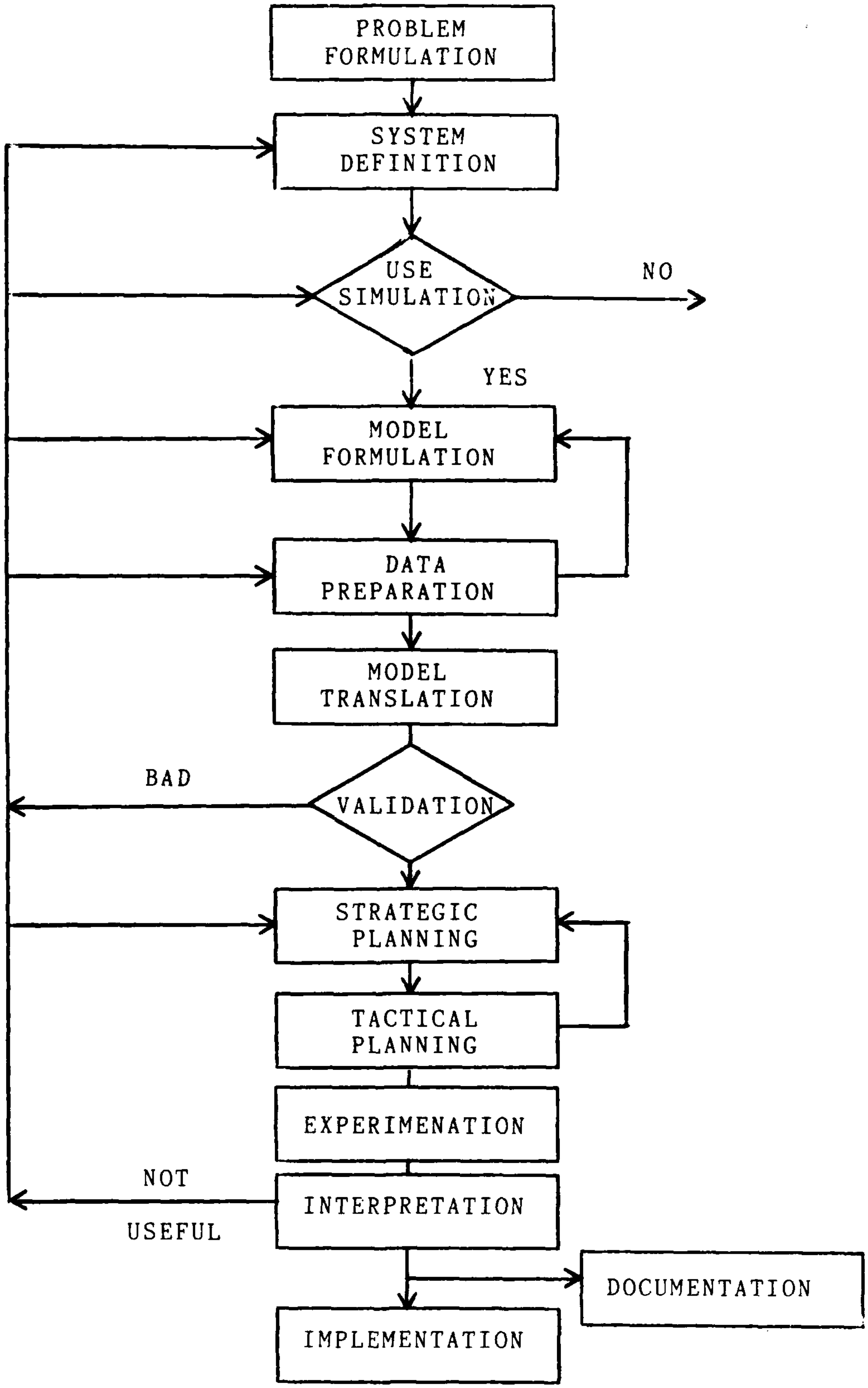

Figure 1. Simulation Process (Shannon 1975)

texts do a poor job in these areas. He states:

> Every study involves data gathering. Data gathering is usually interpreted to mean gathering numbers but the gathering of numbers is only one aspect of data gathering. The system analyst must be concerned with data regarding the inputs and outputs of the system he is studying as well as information about the various components of the system and the interconnections or relationships between them. He is therefore interested in gathering both quantitative and qualitative data, and he must decide what data are needed, whether they are pertinent, whether existing data are valid for his purpose, and how to gather this information. Textbooks usually give the student all of the pertinent information and data without reference to how it was gathered and validated. The student then becomes schizophrenic when faced with his first unstructured problem for which he must determine on his own what data are needed and how to gather them.

It was this reality which motivated this research. It was first felt, that while better and more structured procedures were needed, the primary solution would be more knowledge and experience. However, continued research soon showed that people experienced in the field were identifying poor methodology and procedures as a problem and were looking for a better way to conduct simulation studies.

Problem -- A Broader View

Beginning in the late 1970s there was acknowledgement in the field that recurring problems were the result of poor or inadequate procedures. It was "apparent no one is worrying about the total simulation methodology" (Huhn and Comer 1981). Furthermore, "Current simulation techniques are highly dependent upon the experience and skill of those

applying them and do not provide a good communication tool". (Evers, Backert, and Santucci 1981), and "The primary difficulty with computerized models is traceable to the lack of discipline and control in the development stages and a reliance on nonexistent or outdated planning and project management aids" (Nance, Mezaache, and Overstreet 1981).

Additionally, Miller and Pare (1986) point out that trends in the use of simulation "demand new and more sophisticated techniques to execute and manage modeling projects." These included the trend for simulation systems to support multiple phases of a project development and a more extensive use of a model to support decision-making. This in turn resulted in longer life spans for simulations. Another trend was for the "user" of simulations to be "a person with functional expertise in the area being modeled but no simulation or programming background." This requires the system to be very user-friendly as well as have "automation of the statistical analysis and data base management."

More specifically, Zeigler (1979) stated that "contemporary simulation methodology is inadequate for handling large-scale multi-faceted systems" and identifies three "limitations of conventional simulation techniques":

1. Impediments to facile communications: man-man, man-machine -- Models are difficult to construct for the time being, in part because the computer offers too little help in developing models and verifying simulation programs.

2. Inadequate conceptual framework -- The distinct functional elements involved in

simulation are not clearly distinguished by conventional simulation languages.

3. Lack of necessary software tools and data structures for organizing models and their data -- Facilities for subdividing the model into modules and for organizing them hierarchically to form a working model are necessary to speed the construction of new models from existing ones.

Nance, Mezaache, and Overstreet (1981) state that "difficulty stems from the missing ingredient: model management". Huhn (1981) says "A preoccupation exists in the selection of a simulation language before the system problem and its model are understood. The classic problem is in the attempt to use a simulation language to express the system model, and the basic problem is "there is no established methodology for characterizing a system as a model".

Whatever the reason, though, analysts and simulations were not providing useful results (Annino and Russell 1979; Huhn and Comer 1981; and Nance, Mezaache, and Overstreet 1981). Annino and Russell (1979) pointed out that the key "to successful use of simulation analysis lies in understanding and applying new methodologies." The next section, therefore, will look a little closer at some of these "new" methodologies.

Methodologies

Methodologies are "bodies of concepts, approaches, and methods" which seek to not only define what has to be done but how to go about doing it (Zeigler 1979). For example, a

step in the simulation process is to describe or define the system to be modeled. A methodology will give you a way to do this. Tools or computer software will, in turn, make this methodology easier to accomplish.

In briefly discussing this topic we'll use four of five were categories of approaches suggested by Nance (1979). These "(1) extension of software development techniques,... (3) extension of SPL (Simulation Program Languages) definition, (4) system specification languages, and (5) model-based methodology."

The first has to do with the idea that the output of a simulation study is software. Thus, the programming and software development procedures and tools offer all that is necessary (Huhn and Comer 1979; Miller and Pare 1986; and Golden 1985). These references actually do more than just advocate the use of software techniques to aid in the development and management of simulation projects. They are realistic enough to realize it will take a combination of techniques. Huhn and Comer advocate an Integrated Software Methodology (ISOMET) which provides "a collection of integrated policies, procedures, standard practices, and guidlines which provide increased productivity and management of the software development process." Miller and Pare (1986) view the process as software development but have developed a methodology based on software development techniques and simulation techniques based on Shannon (1975), Nance (1981), and Huhn and Comer (1981). While such techniques come under

some minor criticism from the simulation community, they do attempt to offer a more structured approach to the simulation process.

The SPL approach involves using the simulation language as a framework for modeling the system. This approach has been a great aid in simplifying the task of programming the system model but also has resulted in the system model being constrained by the world view of a particular language (Nance 1983). Nance cites the fact that the number of models programmed in higher level languages such as, FORTRAN, and Pascal, "emphasizes the perceived difficulties of translating modeling concepts into a correct SPL representations."

The system specification languages, or more broadly SPL's are "based explicitly on systems theoretic concepts and the development of conceptual and mathematical theories for guiding the practice of modeling and for designing software tools..." and offer advantages over current approaches (Nance 1983; Zeigler and Oren 1979). For simplicity, the fourth approach, the model based approach, will be included in this discussion because both are predicated on separating the "functional elements" of conventional simulation programs into autonomous modules so each can be worked with separately.

Both also depend on an extensive database management system in which data regarding these functional elements are stored. Zeigler and Oren (1979) categorized the data requirements into four different files: "(1) experimental

frame files" which includes "initial conditions, end-of-run criteria, and output variables; (2) model file which includes "model structure and model output specifications so you can combine a given model structure with different model outputs to have different models of the same system"; (3) model data files which contain the data collected from the computer runs; and (4) the Real System data file which includes data gathered from the system under study to aid in model building (Standridge, 1981; Zeigler and Oren, 1979).

Similarly, Nance (et al. 1981) outlines the components for a model management system. They are "(1) a data base management subsystem, (2) an extensive dialogue module providing the vehicle for producing a communicative model from a conceptual model, (3) a software development subsystem, (4) a documentation production subsystem, (5) an experimental subsystem, and (7) an internal monitoring and accounting subsystem." Again, of prime importance is the "functional partitioning".

In 1983 Nance (1983) discusses this concept in terms of of a Model Development Environment (MDE) which would "provide an interactive setting for model creation so that the modeling activities, supported by necessary model development tools, contribute to long term organization assets in the form of models, data, experimental designs, and experimentation results." Whether this is an extension of the model management system is not known, but it does provide a simple statement of the type of computer support required of the

system or model-based methodologies. The next section will look at this aspect.

Simulation Support Systems

This section will look at simulation support systems and focus on integrated support systems as defined in chapter 1. As an example we'll look at The Extended Simulation System (TESS - Pritsker and Associates).

Standridge (et al. 1985) describes four generations of simulation software. The first generation includes "languages such as Q-GERT and GASP II which provided a single world view for constructing models." The second generation involved extensions of the first, and allowed for more than one world view in a single language. Examples of this type of software includes SLAM and GASP IV. The third generation "recognized that software was needed to support other aspects of a simulation, beyond the model conception and implementation provided by simulation language." Examples here include AID and UNIFIT for fitting input data to distributions, SDL for managing the data of simulations, and SIMCHART, SEE WHY and SIMAN for graphical presentation of simulation results.

In a separate article, Standridge (1985) points out that applications showed "the value of having support systems" and also "established the usefulness of separating the analysis and presentation of simulation results from the simulation run as well as separating the analysis results from their presentation. In addition, benefits of data base management

techniques to organize and select simulation model inputs and results were established. The ability to automatically collect data during simulation runs was established. In presenting simulation results, especially to nonsimulation experts, the usefulness of graphics (including the animation of simulation runs) was demonstrated."

These concepts are really the basis of the methodologies described above, and an integrated environment incorporating these concepts represents the fourth generation. TESS is an example of this type of software. It would also seem appropriate to include some of the interactive graphically animated systems, such as, Cinima/SIMAN, AUTOGRAM, and RTCS (Grant and Weiner 1986). However, TESS, and to a certain extent SIMPLE_1 purport to be integrated systems. TESS will therefore provide the basis of the remainder of this discussion.

Table 1 outlines the specifications upon which TESS is built (Pritsker 1986; Standridge 1983; and Standridge 1985) and Figure 2 shows the TESS architecture (Standridge, Vaughan, and Sale 1985). Essentially, "TESS provides a framework for performing simulation studies" (Standridge, 1985). It is described as a generic system and can currently support SLAM II, MAP/I, and GPSS/H simulation languages (Pritsker 1986). To briefly show what TESS can do, table 3 lists some of its capabilities (Standridge, et al. 1985).

TABLE 1

SUPPORT SYSTEMS SPECIFICATIONS

A Simulation Support System:

1. Must support model building.

   - Must provide an environment for the analyst to remodel.
   - Models should be easily recalled and categorized.
   - Model building should be interactive and graphical.
   - A documentation trail regarding model development should be maintained.

2. Must support the data management tasks, to include storage, retrieval, and organization of system data, experimental control specifications, and simulated generated output.

   - The simulation outputs should reference both the experimental specifications and the model from which it is generated.
   - Procedures are required to assess, edit, concatenate, and display data stored in the data base.
   - It is also useful to be able to assess the simulation outputs for presentation in spreadsheets or as inputs to other models.
   - Once in the database, the form of the simulation data should not differ from the form of the actual system data.

3. Must support analysis and reporting tasks.

   - It is necessary to estimate the dry-up or close-down time for system operation.
   - Procedures for interrogating data obtained from a simulation run are needed to explore such situations within runs and over multiple runs.

(Pritsker 1986; Standridge 1983; and Standridge, Vaughan, and Sale 1985)

TABLE 1 -- Continued

- Support should be provided to transform automatically the outputs of the simulation runs into inputs for statistical analysis procedures such as regression analysis, analysis of variance (ANOVA), auto-regression time series programs, and curve fitting programs.
- The support system should provide the capability to output the data in both graphical and tabular form.
- The system should be capable of working with different output devices.
- Procedures need to be included to easily define the type of data to be input into the data base.
- For system data, a convenient mechanism is necessary to support the data definition process.

4. Must support the visualization task using animation.

   - The system should provide the tools to convert either system data or simulation output data into a form which portrays the changes on a facility diagram.
   - The visualization process should be able to run in post simulation mode or in a concurrent simulation mode.

5. Must support all aspects of the simulation project.

6. Must tailor its simulation results to the needs of the decision-maker.

7. Must take a modular approach to simulation projects.

8. Must minimize the need for technical computer knowledge.

9. Must recognize the difference between model builders and model users.

10. Must require the analyst to concentrate on only one form of the model.

11. Must be able to accommodate changes as a result of advance in simulation and computer technology.

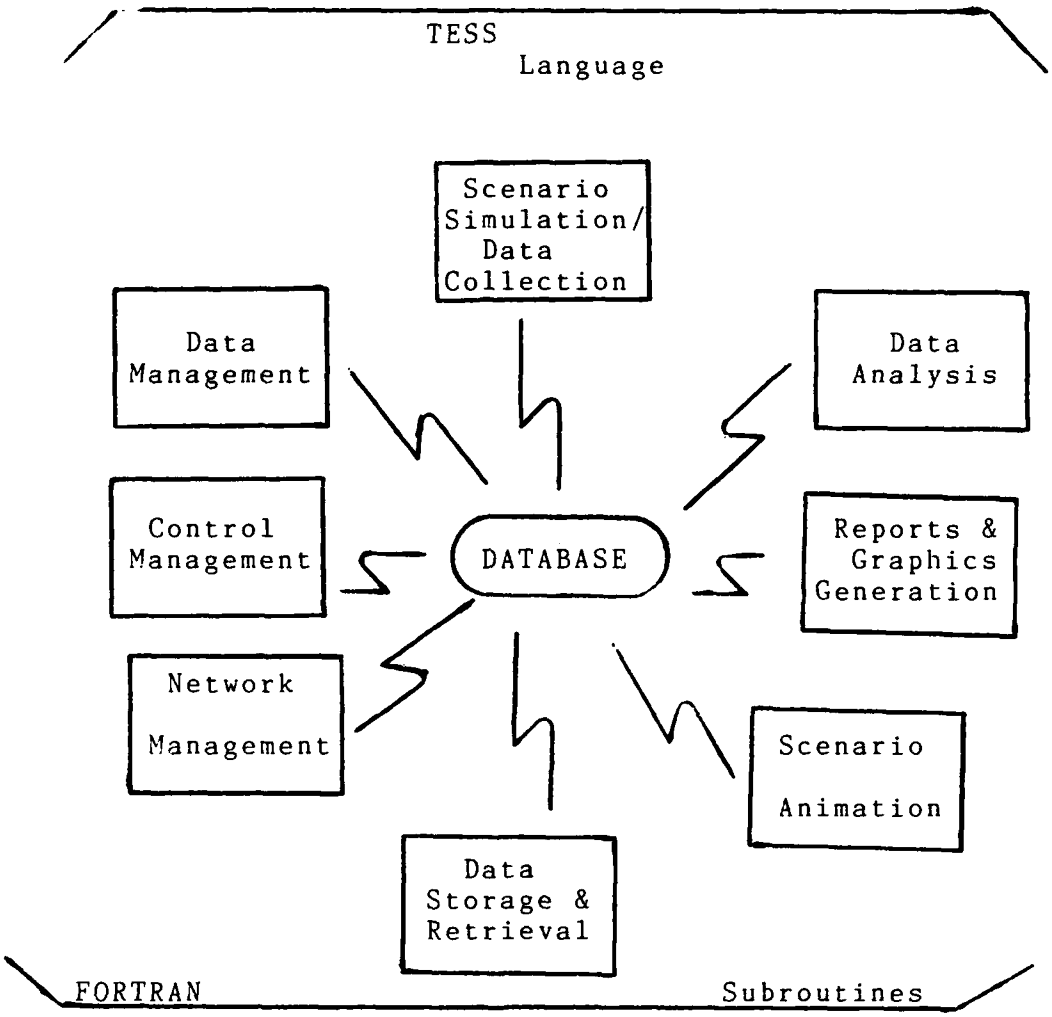

Figure 2. TESS Architecture (Standridge, Vaughan, and Sale 1985)

TABLE 2

TESS CAPABILITIES

TESS Provides:

1. A framework for problem solving using simulation.
2. Separation of the analysis and presentation of simulation results from their generation in simulation runs.
3. Integration of modeling and simulation execution with reporting, graphing and analysis capabilities.
4. A command language to access each capability used in problem solving.
5. Creation and management of network models.
6. Independent specification of experimental conditions for controlling simulation runs (CONTROLS).
7. Management of user defined data.
8. Procedures for combining CONTROLS, data and models to specify alternatives called SCENARIOS.
9. A report generator for presenting simulation results and other data.
10. Graphing of networks, simulation results and user defined data.
11. Procedures for dynamically presenting the operation of a model, that is, the animation of simulation results.
12. Computation of frequency distributions and statistics as well as estimation of confidence intervals.
13. Support for database management tasks.

(Standridge, Vaughan, and Sale 1985)

The Need

Basically, it appears TESS gives the modeler a very powerful tool for accomplishing simulation projects. If you look closely, though, the methodologies and software we've discussed focus on the modeling and the implementation phases of the simulation process. These methodologies seek to move modeling, now described as an art, closer to being a science. Each is predicated on the presence of quality data, both qualitative and quantitative. What is needed, therefore, is a system to support the support system (Figure 3). Something that will aid the analyst in gathering the needed data and information.

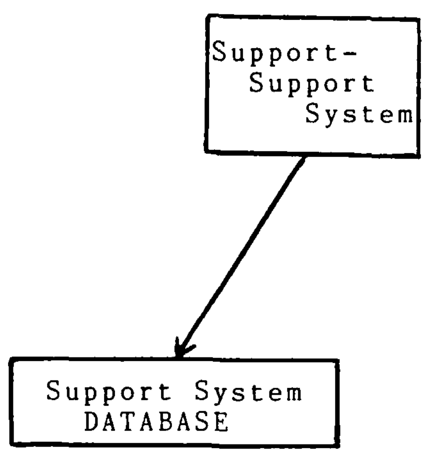

Figure 3. A Support-Support System

CHAPTER 3

THE SIMULATION PROCESS

Before a suitable methodology could be developed it was first necessary to get a better understanding of the process, especially the initial phases. This chapter reviews the procedure that was used in developing a more detailed breakdown of the initial phases of a simulation study.

Approach

In analyzing the simulation process, "structured analysis" was selected as a method by which to provide a more organized approach. Structured analysis (Weinberg 1979) "is a disciplined approach to structuring the system analysts job." Because of the variety of functions and of systems development phases in which the system analyst is involved, structured analysis is defined as a philosophical, top-down approach to all phases of the systems life cycle." This methodology not only addresses the "different phases of systems development in the form of a structured methodology but also the communications and coordination between phases required to make the development a success."

The structured analysis methodology utilizes a bubble type diagram (Figure 4) with the bubble representing an

action and arrows leading in and out representing inputs and outputs for that action. Arrows can also represent required communications.

By considering the simulation process as a system, the

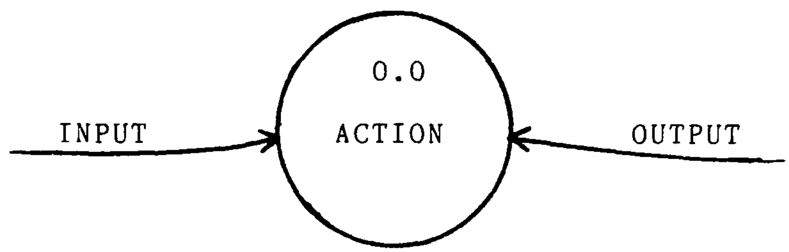

Figure 4. Structured Analysis Diagram (Weinberg 1979)

task was to analyze the system, identify subsystems, and the interactions between them. The idea was to start very broad and general, and work towards a more detailed and specific description.

The Process-General Description

In its most basic form the simulation process is a system with a user problem being the primary input (a reason for conducting the study) and a solution to that problem being the primary output. Available knowledge about the problem and system to be modeled is also a required input. Additional knowledge about the problem and system is another output. Figure 5 represents this basic system (Zeigler 1979 and Shannon 1975).

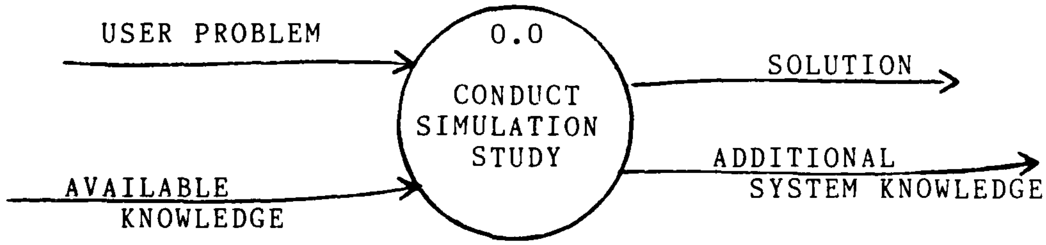

Figure 5. The Basic Simulation Process

The success of this process as well as the way it will be further broken down is predicated on several concepts discussed by the above referenced sources and others (see reference section). First, and most obvious, is that the quality of the simulation process is only as good as the inputs, the simulation model, and its output. This really

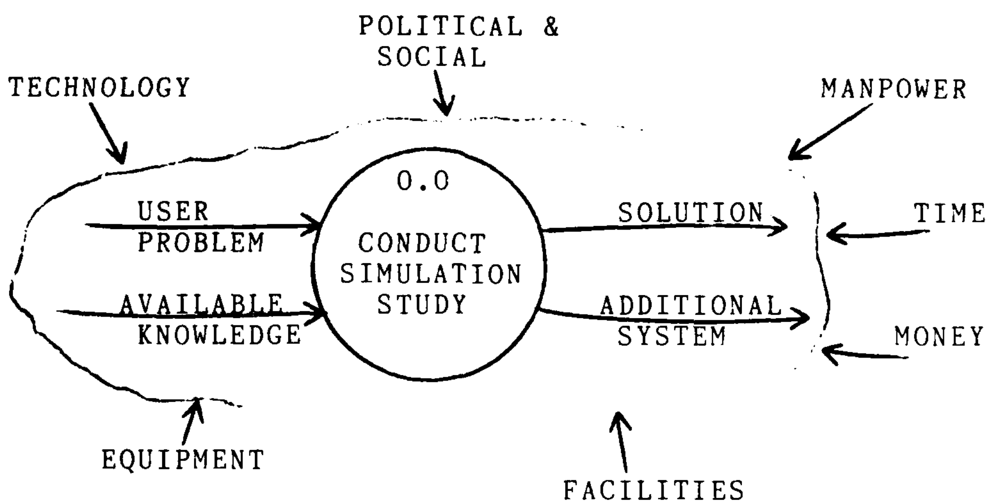

Figure 6. Influence of External Factors on The Simulation Process.

entails the quality of input and output data, and the validity of the model.

Second, a simulation process is a project which must be managed and like all projects is subject to various constraints and external factors (see Figure 4) (Kerzner 1984). These external factors are important because they may impact on the feasibility of using simulation and/or the acceptance of the study, irrespective of its quality. The goal should therefore be to internalize these factors into the process.

Third, the simulation process, in Figure 1 and other sources, is normally described as an iterative process. Pritsker (1986) states "The stages of simulation development ... are rarely performed in a structured sequence beginning with problem definition and ending with documentation. A simulation project may involve false starts, erroneous assumptions which must later be abandoned, reformulation of the problem objectives, and repeated evaluation and redesign of the model. If properly done, however, this iterative process should result in a simulation model which properly assesses alternatives and enhances the decision-making process."

Finally, now that this third concept has been stated, is the concept or thesis that if many of the problems common to simulation studies are to be avoided, a more structured approach must be taken. It does not seem reasonable to start model development without a clear and accepted definition of

the problem and the objectives to be attained. Also, the system to be modeled must be properly defined. Certainly, whatever approach is taken, it must be flexible. Every problem solution method is iterative in that you are continually discovering more facts and data. (This concept implied but never actually stated in the following: Barnes 1968; Evers, Bachert, and Santucci 1981; Miller and Pare 1986; Martin 1968; Huhn and Comer 1981; Schmidt 1985; Weinberg 1979).

With these concepts in mind the next step was to start defining subsystems beginning with a structure which resulted in some sequence of actions with the output of one subsystem

TABLE 3

PHASES OF THE SIMULATION PROCESS
ACCORDING TO BANKS AND CARSON

Phase	Steps
1	1. Problem Formulation 2. Setting of Objectives and Overall Design
2	3. Model Building 4. Data Collection 5. Coding 6. Verification 7. Validation
3	8. Experimental Design 9. Production Runs and Analysis 10. Additional Runs
4	11. Documentation of Program and Report Results 12. Implementation

(Banks and Carson, 1984)

representing the input of the next subsystem. Martin (1968) defined the process as three phases; developing the conceptual model, model implementation, and model results. Banks and Carson (1984) divided their 12 step process into four phases as shown in Table 3.

Combining these concepts and ideas from other sources the system was expanded into six subsystems, or phases, (see Figure 7). While ideal in nature, the output of each phase is the input of the subsequent phase. This further implies that a phase must be completed prior to beginning the next one. Also, while the iterative nature of the process is not shown, this concept is evident within each phase.

The process begins with an investigation and analysis of the problem. The outcome of this phase is knowledge about the problem and system. This includes (ideally) an explicit definition of the problem, explicit goals and objectives to be accomplished by the study, constraints and limitations, information and data defining system operation, and a static definition of the system. This data should be validated with the user/sponsor and a preliminary cost/benefit analysis conducted. All key decision-makers, system experts, or others that may influence the study or its outcome should be identified, along with sources of data and information.

As shown in Figure 7, a detailed investigation of the problem and system may result in an acceptable solution to the problem. Not shown though, is that this phase is really carried on throughout the entire process. In a more

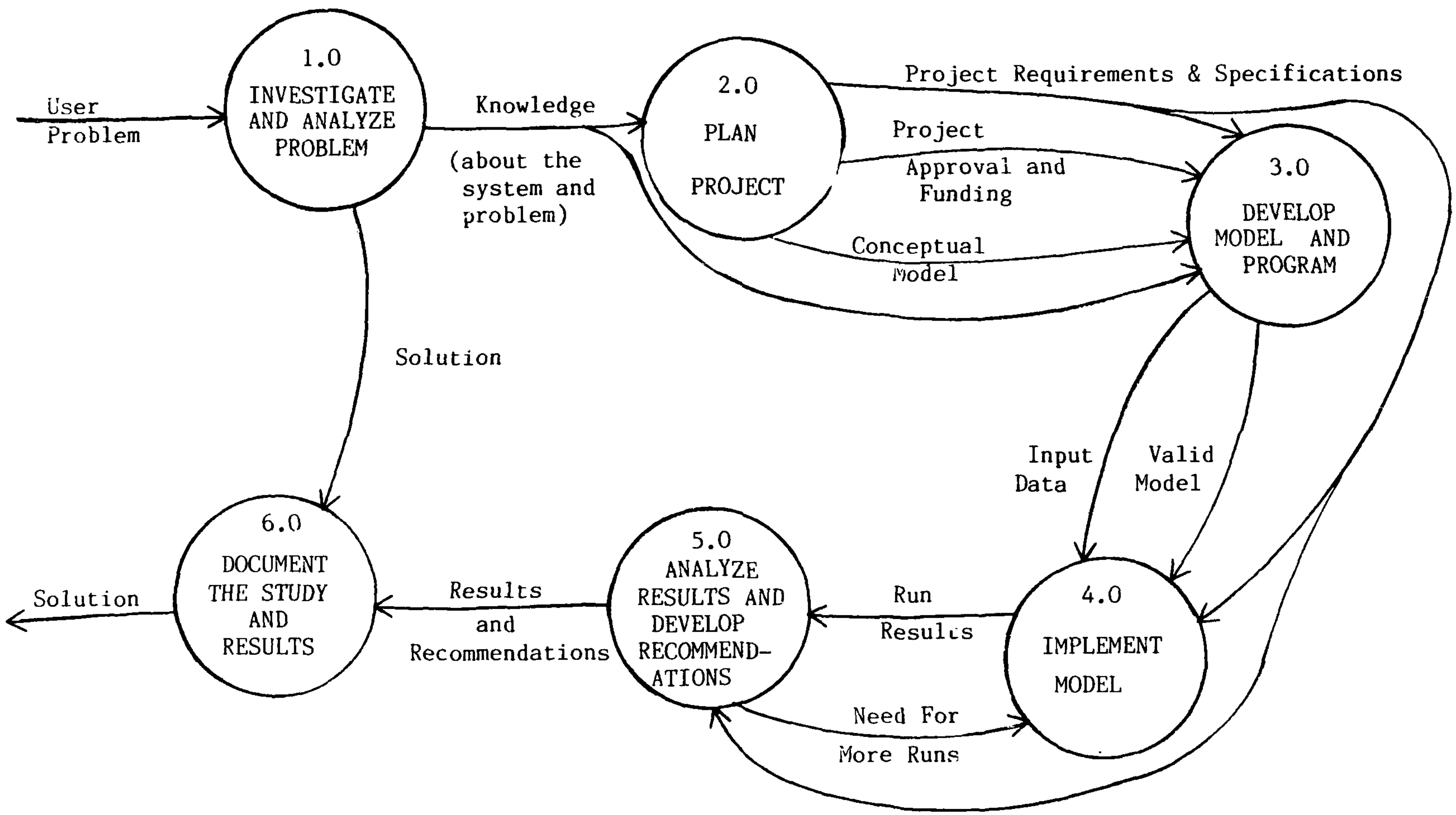

Figure 7. Flow Diagram -- Simulation Process

structured sense, this phase may be divided into two phases; a preliminary investigation prior to project planning and a more detailed investigation following project approval when funding and additional manpower may be available. The preliminary goal, however, is to complete the phase prior to beginning extensive modeling and programming.

Phase 2 involves using the knowledge gathered in phase 1 to develop a conceptual model then develop and evaluate alternatives for how to implement this model. Then having decided that simulation will be the appropriate modeling methodology, a detailed plan and proposal for how to conduct this study is developed and presented to the user/sponsor. This insures that the problem and objectives are understood both by the user and the analyst, each understands what will be expected from the other, and what will and will not be achieved by the study. Also, the time and cost constraints are accounted for and a detailed cost/benefit analysis completed. Finally, detailed project requirements/specifications should include preliminary planning for how the model will be implemented, strategic and tactical planning, and experimental design.

Phase 3 is the actual development of a model which is translated to computer code. It is during this phase, after input data requirements have been firmly defined that this type of data is collected. The output of this phase is a valid model, accepted and understood by the user/sponsor.

Phase 4 is where the model is actually implemented.

Detailed experimental design, and strategic and tactical plans are completed and simulation runs conducted. It is here, as well as with the collection of input data, that additional trade-offs may have to be made because remaining funds and time may limit the number of experiments that may be conducted. This may result in a lower level of accuracy and detail. It is therefore important for these issues to be considered during the project planning phase and accounted for in the project budget (Shannon 1975).

Phases 5 and 6 involve analyzing the results and developing recommendation for the user, and preparing documentation of the study. The ultimate outcome is a solution for the original problem.

Utilizing the structured analysis technique each phase was further subdivided with the level of detail increasing at each successive level. Again, the purpose of this breakdown was to develop an understanding of the simulation process and put into perspective what goes on in the early phases of the simulation process; primarily subsystem, or phase 1, figure 7.

Specifically, the tasks, data to be collected and archived, and sources for this data needed to be identified. The results of this analysis, as it relates to the two major data outputs of phase 1, problem definition and system definition, are summarized in Tables 4 and 5. This information was then used to develop a generic "support-support" system specification which will be discussed in the next chapter.

TABLE 4

PROBLEM FORMULATION SUMMARY

TASK	DATA	SOURCE
Schedule Interview	Key Decision-makers	Project Sponsor
Plan Interview	Key Managers	Key Decision-makers
Conduct Interviews	Users/Experts	System Experts
Conduct Research	Important Sources of Information	Documents
Observe System	Communicated Problem	Observing System
Operation	Operating Policies	Participating in System Operation
Analyze Data/Info	Perceived Benefits of Solving Problem or Obtaining Additional Information	
Prepare and Submit Periodic Updates to Users for Validation	Objectives (of system and key decision-makers and managers	
	Criteria to measure Objectives	
	Constraints/Limitations On System	
	Constraints/Limitations On Study	
	Expectation of Users	
	Factors Affecting System	
	Assumptions/Hypothesis	

TABLE 5

SYSTEM DEFINITION SUMMARY

TASK	DATA	SOURCE
Schedule Interview	System Environment	System
Plan Interview	Environment Factors	Documents
Conduct Interviews	System Components &	Users/Experts
Conduct Research	and Subcomponents	Managers
Observe System Operation	Inputs and Outputs for These Components and Subcomponents	Documents
		System Designers
Develop a Schematic Static Represent-ation of System	Assumptions and Hypothesis	Participating and Observing System Operation
Present Findings to Users and Validate system Definition		

CHAPTER 4

GENERIC SPECIFICATION

This chapter will outline the specification for a generic "support-support" system. The first section will discuss the general concept of a support-support system. The second section will identify the major functions such a system should support, and the last section will outline general system requirements.

To simplify terms, the remainder of this report will refer to the support system as discussed in Chapter 2 (TESS) as the primary support system. The system being proposed will simply be referred to as the support system.

Basic Concept

Based on the breakdown produced in Chapter 3, the proposed support system should support all aspects of phase 1 and task 3.4 in phase 5 (collect input data). The primary support system as defined in Chapter 2 appears to be capable of supporting phases 3-6, with the exception of 3.4, and most of phase 2.

The primary purpose of this support system will, therefore, be to support the data collection aspects of a simulation study. Detailed analysis of this data will occur in the

primary support system. However, the proposed support system needs to provide for enough analytical ability so the analyst can adjust the collection effort as problems arise or new information becomes available.

In general, the support system is to help in:

1. organizing and guiding the collection effort;
2. organizing and archiving the data;
3. displaying the data for review, analysis, update, and validation; and
4. Producing required reports.

The goal of the proposed system is to insure the data (both subjective and objective) necessary to perform a simulation project is gathered in as efficient and effective manner, and the problems common to many projects be avoided.

Major Functions

The proposed support system should provide help in four major areas and two minor areas as shown in Figure 8. The major functions consist of problem definition, system definition, input data collection, and project planning and management. The minor functions support the other four and are economic analysis and report generation. This section will briefly discuss these 6 functions.

Formulate Problem. The support system should aid in the orderly collection of data necessary to define the problem. Table 4 shows the tasks and information applicable to this function.

The problem definition will consist of 4 issues. The

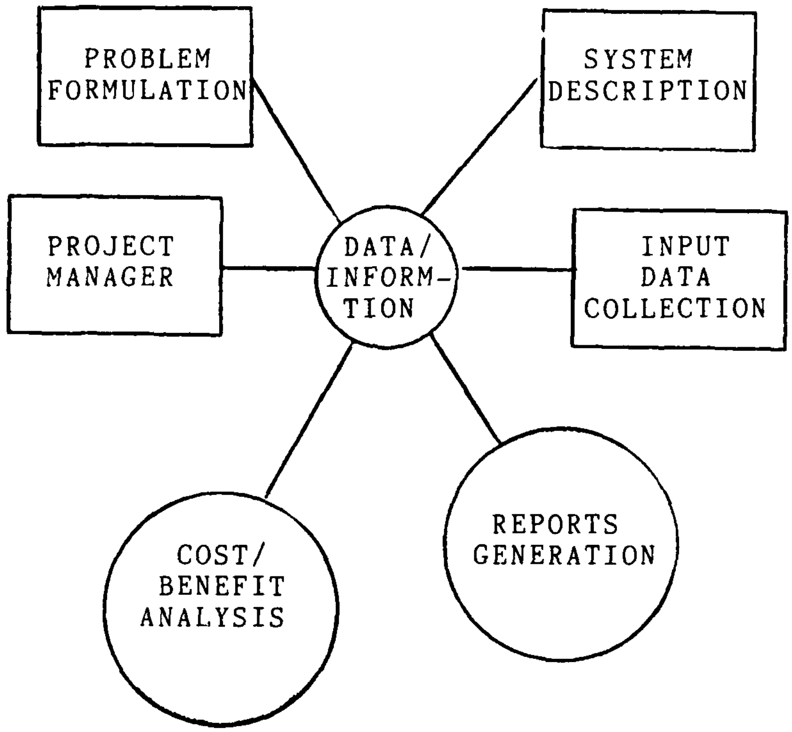

Figure 8. Support-Support System Architecture

first is a workable definition of the problem as communicated to the analyst by the users. Often times the only thing people are sure about is the fact there is a problem. The analyst will start with a list of symptoms and have to identify the underlying causes. The second is to take into account the politics of the organization as it relates to the problem and its solution. That is why it's important to identify and interview all key people who can add to the understanding of the problem, and also affect the implementation of the solution. Third, is to determine the benefits that will result from solving the problem. This is closely

related to the fourth issue. What it will take to solve the problem; both in terms of methodology, manpower, and money? Normally, if the cost of solving the problem is more than the benefits, there is little reason to continue the project.

The data will be collected from numerous sources, but the primary method will be by interviewing key decision-makers, managers, and system operators (experts). Other sources include observation, and researching applicable regulations, procedures, manuals, and policy statements. Of interest also, is whether any previous studies have been accomplished concerning the problem. These previous studies can provide valuable insight into both the problem and politics of solving the problem.

System Description. The system should aid in developing a thorough and accurate understanding and description of the system to be modeled. Table 5 shows the tasks and data associated with this function.

The support system should provide various methods to schematically represent the system. This is necessary because different systems require different methods and different analysts may prefer one method over another. Also, a combination of methods may be required to adequately describe the system operation.

The support system should provide a format for a complete verbal description of the system operation and its respective components, to include inputs, outputs, and factors influencing its operation.

Whatever methods are used, it should be compatible with or translatable to the modeling representation used by the primary support system. However, the primary function is to obtain knowledge about the system and have it depicted and described in such a fashion that this knowledge can be understood and validated by the user's/sponsor.

Input Data Collection. The system should aid in developing a data collection plan and provide formats and a means for collecting this data. The primary support system, through the modeling process, will define what data is to be collected, but the proposed support system should help in planning where, when, and how to collect it. It should also provide a means to archive this data and aid in inputting it to the primary support system. This system should provide for preliminary analysis of the data to insure it meets the statistical assumptions of independence and homogeneity. Finally, the system should provide a means to collect data by the observing system operation and/or by inputting data from documents.

Project Planner. The system should support the required planning tasks, both personal and for the project as a whole. The project planner must also help in accounting for resources expended to accomplish the project. The support system should also help in developing project cost estimates.

Report Generator. This function will support the other five. The system should provide help in developing all reports necessary during the initial phases of the project.

Standardized formats should be developed which permit the data to be entered as it becomes available. However, the system must provide the flexibility to adjust to requirements of the particular project or analyst.

Economic Analysis. The system should help in analyzing the cost/benefits of solving the problem. Coupled with the project planner function, the system should help in making many of the trade-off decisions that arise during the course of a project. This includes the amount of data to be collected. Is there a point at which the cost of collecting more data/information overshadows the benefits?

General Support System Requirements

The support system should support the iterative nature of this phase of the process. While the simulation process as a whole is an iterative process, the early investigative phases are particularly so. Each source of information leads to another.

The support system should be able to adapt to the nature of the particular project. Each problem and system are different. While the support system calls for a more standardized and structured approach, it must be flexible.

In the same way, the support system should take into account the various approaches used by different analysts. Whenever possible, the support system should provide multiple formats and techniques for gathering the information.

Data collected in support of one of the above listed

functions but applicable to one or more of the other functions must be identified as such and cross-referenced. This is to prevent the analyst from collecting the same data twice.

Finally, the support system should insure the analyst considers all information and data requirements. Each project is different so some of the data requirements will not be applicable, but the analyst should make this decision and simply not overlook it. Also, information/data requirements which are applicable but not yet defined, should result in assumptions and hypothesis concerning the data. Then the analyst can continually review and update these assumptions as information/data becomes available. These assumptions/hypothesis can, in turn, help the analyst develop new questions and avenues of investigation.

CHAPTER 5

CONCEPT FOR IMPLEMENTATION

This chapter will examine how this specification can be implemented using a portable microcomputer. The first section will discuss why a portable microcomputer should provide a suitable platform. The second section will suggest how the functions defined in Chapter 4 can be implemented.

For the most part this implementation strategy is still conceptual. As stated in Chapter 1, this study resulted in only partial implementation of the input data collection function which will be discussed in more detail in the next chapter.

Why Use A Portable Microcomputer

While a workbook or guidebook which provided the applicable checklists and formats would be a possible alternative, a computer can provide the same features. Most important, the purpose of the system is to collect and archive data. The computer is ideally suited to this task. The computer will also provide the flexibility to change collection and report formats to meet the needs of the project. Finally, this can all be done with a computer you can fold-up and carry with you to the collection site.

Portable microcomputers, specifically, the Radio Shack

Model 100, and special electronic data collectors have been in use for some time, particularly for work measurement, work sampling, and quality and inventory control (Savage and Keevan 1984; Wilkerson 1984; and Martin 1984). These devices especially when combined with the office PC, have reduced the time and effort to conduct these studies (McDermott 1984; Dossett 1984; Sprague and Schoten 1984; MacMillan and Walker 1985; and Wilkerson 1984).

Dossett (1984) points out that these collection devices do have limitations, though. They do not have the power to develop detailed summaries on multiple studies, and are quite slow. Also, archiving data on cassette tape is a problem. They can perform certain statistical tests, but are slow and limited in what they can do. Furthermore, these devices don't have the keyboards and screen displays for more powerful editing.

Since Dosset first made those observations, the portable data collectors have improved. They are faster, and can perform more statistical tests. They are small and can port their data to larger computers for analysis. They are limited in the type of data they can collect - numeric data.

Though not hand-held, some of the newer portable microcomputers are small and light enough to be called "laptop" and have capabilities equal to the desktop PCs; up to 640K of RAM, dual disk drives, full 80X25 character/line displays and full keyboards. These qualities make up for the limitations of the hand-held devices and can provide the capabil-

ity to support all aspects of the data collection process.

One drawback at this time is the lack of readily useable software. Most of these computers use the 3 1/2 inch disks. They can hold up to 720K data but there is not as much software available in the 3 1/2" disk size. This should only be a temporary problem, though.

Implementation Strategy

Ideally, to implement this system, a single software package which integrates all the required functions is necessary. As a starting point, however, this study suggest that the required application programs, consisting of standard formats, be developed using generic software, such as, spreadsheets, database managers, word processors, and programs written in BASIC. Whenever possible, utility programs should be written to integrate the different functions.

These software packages are readily available and most potential users are, at least, somewhat familiar with their operation. Also, they provide the user the maximum flexibility to adjust them to his/her own desires and/or the particular situation.

Application programs should be run in drive A and a data disk in drive B. To avoid mixing data, a data disk(s) should only contain data from one project. Also, the concept envisages the need for numerous data files. Using a 720K 3 1/2" disk, the user is limited to 112 files. If subdirectories are used then the number of files are limited only by

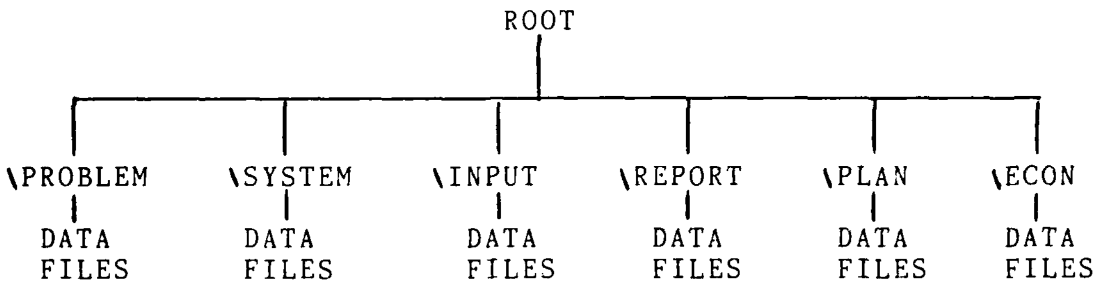

Figure 9. Data Disk Directory

the amount of memory on the disk. A tree type file directory is therefore recommended (Figure 9).

The remainder of this section provides some brief suggestions on how each system function could be implemented.

Formulate problem - Utilizing a word processor or data base manager, formats should be created to lead the analyst through the process of collecting data. A data base manager, dBASE III would appear to provide the most flexibility, because the programmable features of DBASE III would allow the analyst to input both the formats and the logic to guide him/her through the process. Database files could be created for each data type category.

To provide a basis for building the formats and logic, Balci and Nance (1985) offer a very good methodology which lends itself to computerization. As long as the problem can be categorized as descriptive or prescriptive, the analyst follows a list of questions/information requirements which, depending on the outcome of each step, guides the analyst to another appropriate question/information requirement. Figure 10 shows a flow chart which depicts the procedure, and

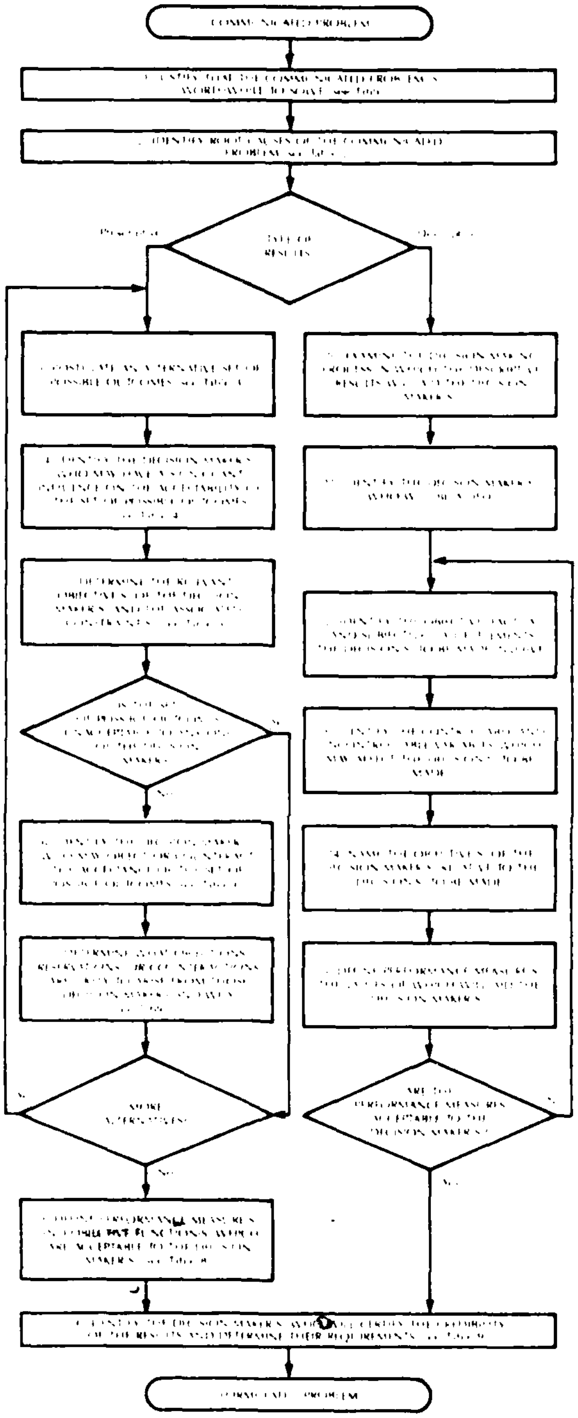

Figure 10. Problem Formulation Methodology (Balci and Nance, Figure 1, 1985)

TABLE 6

JUSTIFY THAT THE COMMUNICATED PROBLEM IS WORTHWHILE TO SOLVE

<1a> If it is perceived that a set of current conditions deviate from a range of acceptable conditions or a desired set of conditions, go to <1d>

<1b> If a need is perceived to obtain some required information for decision making, go to <1e>

<1c> If a set of conditions reflecting no significant deviation are sought go to <1f>; otherwise go to <1g>

<1d> Is this deviation significant? If not go to <1h>. Does the comparison of potential benefits of correcting this deviation with the estimated cost of correcting it justify an attempted solution? If not go to <1h>; otherwise go to Table 2.

<1e> Does the comparison of potential utility of this information with the estimated cost of obtaining it justify obtaining this information? If not go to <1h>; otherwise go to Table 2

<1f> Does the comparison of potential benefits of this set of conditions with the estimated cost of achieving it justify the attempt to obtain this set of conditions? If not go to <1h>, otherwise go to Table 2

<1g> Examine the context of the communicated problem and reexamine the benefits/cost (B/C) ratio to justify a solution attempt. Go to Table 2.

<1h> The problem is not worthwhile to solve. The solution cost is likely to exceed the return. Terminate

(Balci and Nance, Table 1, 1985)

TABLE 7

IDENTIFY ROOT CAUSES OF THE COMMUNICATED PROBLEM

<2a> Examine the symptoms described within the communicated problem and analyze causality relationships within the context of the problem environment

<2b> List and label all the symptoms, problematic situations, problems, factors, and conditions that affect each other in causing the communicated problem.

<2c> Construct a causality network by drawing a series of edges crossing the labeled elements in <2b> to represent how they relate to each other. (One can contribute to another, be caused by another, or be independent of another.)

<2d> Identify the root causes as the ones with no indirected edges

<2e> If the communicated problem requires a prescriptive solution, go to Table 3. If it requires a descriptive solution, go to Stage 10

(Balci and Nance, Table 2, 1985)

Tables six and seven show examples of the steps the analyst would follow to accomplish each phase of the procedure. Refer to Balci and Nance (1985) for additional information. The article also describes a questionnaire type procedure for verifying the formulated problem.

Not included in this procedure are the data categories of assumptions and hypothesis. These would be added with the logic guiding the analyst to make assumptions and hypothesis regarding data that is not yet known. Then as information is collected, the analyst will update the assumptions. The goal here is to insure the analyst considers all the data requirements.

System definition - This function will involve a verbal description of the system, plus a means to schematically depict it. Various formats for doing this should be available to the analyst. Shannon (1975) suggests process charts, flow diagrams, activity charts, and block and logic diagrams, as possible ways to help describe the system being modeled. Pictorial graphics is also an option. Whichever way is used, the system should have a word description file associated with each schematic, so as the schematic is created or analyzed, the details of its operation can also be entered/reviewed.

Project planner - Numerous project planning software packages are available. The choice of which one to use would be up to the user.

As stated earlier, a simulation study is a project and

must be managed. A project planner would help the analyst do this. It could be used to manage the personal time of the analyst and project staff. It could also be used to estimate and track the project resources; money, manpower, equipment and time.

A standardized project format needs to be developed which will provide a starting point. The analyst can then adjust this format to suit the needs of the particular project. Also, each function should incorporate a utility which logs the time and other resources expended and updates the project planning files.

Economic analysis - The primary focus of this function is to aid in conducting cost/benefit analysis studies of the project. Any of the engineering economy software packages will suffice. However, coupled with the project planner and a decision type model (Gray 1978) this function can help in making many of the trade-off decisions that arise during the course of a project.

Reports generator - Standardized report formats can be developed using the programming portion of dBASE III or a similar data base manager. Then the applicable data can be merged from the appropriate files.

Input data collection - The system needs to support collection from at least two general sources; by observing the system, and from documents.

To aid in collecting data from documents, a program is needed to help the analyst build collection formats. Then

the data can be entered into the computer as these documents are reviewed. The required formats can be determined during the investigative phase. The programmable features of a data base manager could be used. This would provide great flexibility, but the analyst would have to develop a program for each format. Thus, a program needs to be developed that will assist the analyst in building these formats.

Regarding the other source of input data, observing the system, there are many programs and collection devices that support time and motion studies that can be adapted to this function. MacMillan and Walker (1985) developed such a program that provides a conceptual framework for the input data collection function of this support system. The next chapter will review this concept and outline a program that partially implements this specification.

CHAPTER 6

INPUT DATA COLLECTION PROGRAM

This chapter will discuss a program, written in Advanced BASIC, which is conceptually based on a program developed by MacMillan and Walker (1985), but enhanced and tailored to meet the needs of data collection in a simulation study. The first section briefly outlines the concept of MacMillan's and Walker's program. The second section provides a more definitive specification on which this program was built. The third section discusses the operation of the program and the last section will provide a simple example to demonstrate its operation.

Concept

Basically, a data collection device should be based on the concept that the analyst's primary function is to observe. Data entry should be done in such a way as to minimize the time the analyst spends entering data, as this distracts from this primary task of observing.

MacMillan and Walker developed a program, written in BASIC, for use on a Radio Shack TRS-80 Model 100, and is recommended for a starting point. By using the computer clock, the time an event occurs can be logged. They have even developed a routine to access the computer timing

crystal for greater accuracy. When an event occurs the operator hits enter and the time is logged. The operator is then prompted for an event number. This is, in turn, entered and logged. The operator must keep track of event or activity numbers and the logging of time and event identifiers requires multiple keystrokes. Despite these drawbacks, it does offer a good starting point.

Program Specification

Because the system specification developed earlier is rather general and conceptual in nature, this section will list some more definitive specifications.

1. The program should be menu driven and/or the user asked to respond to specific questions; i.e. easy to use.

2. Data entry should be limited to a single keystroke to the maximum extent possible. All information which describes the specific event or activity should be entered prio to beginning data collection. This is so the analyst can maximize the time spent observing, and thus maximize the number of events/activities he/she can track.

3. The computer display should define or display data to be collected and aid in tracking the collection process. Put another way, the display should somehow define what activities to observe, how to enter data for this activity, and what data has been entered.

4. All data file operations should be invisible to the user. The computer should manage the files, not the user.

5. Data files should be identified by project,

scenario, study number and activity number. The project number is a name or number which ties the data to a particular simulation project. A scenario defines a specific set of activities or events. A study number defines one set of homogeneous data collected for a particular scenario. For example, if data are collected on a particular activity over a period of several days, each days data would be associated with a particular study number and kept in a separate file. This is so the data is not merged until it can be statistically analyzed to determine if it is homogeneous.

6. The program should also collect data and calculate statistics that supports model validation. While this is more a function of properly defining the proper data requirements (a function of the primary support system), the collection program can help by keeping track of queue size, etc.

7. Data generated in this program will be filed under the ±INPUT directory.

Program Development and Operation

The program is predicated on the concept that the modeling process accomplished by the primary support system will determine what inputs are required to run and validate the simulation program. What the support-support system will do is help the analyst to plan how the data will be collected and a means to collect it (see Figure 11 for general logic diagram). The first step is to develop a plan.

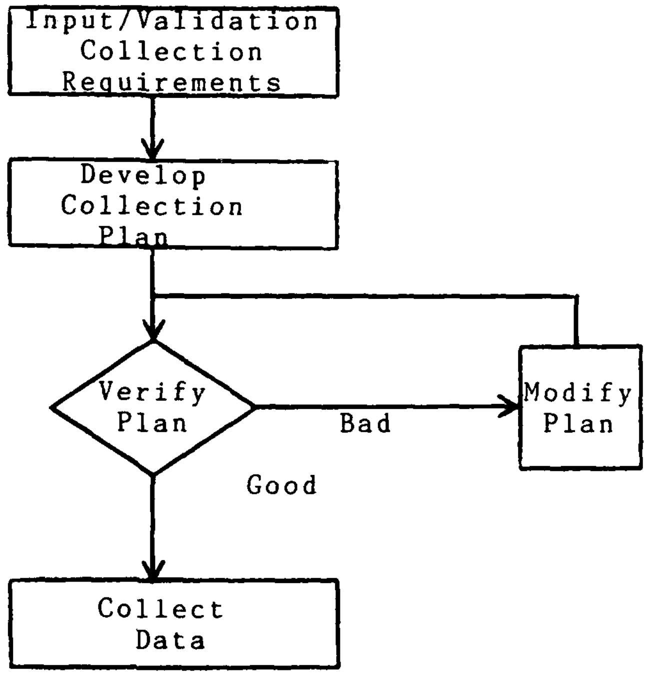

Figure 11. General Program Logic

Developing the Project Plan

The plan is developed by building the required collection scenarios. The scenarios, as defined above, are the activities/events an analyst will observe from one location. How many activities one analyst can observe and collect data on from one location depends on the physical layout of the system, the complexity of the activities/events, and the skill of the observer/collector.

This program allows the analyst to collect data for six distinct activities. If there are similar activities occuring at the same time (termed parallel activities in this program) the program will collect data on up to 3 parallel activities for each distinct activity. Thus, one scenario

may actually collect data on up to 18 activities/events.

The collection plan is defined by a simple schematic composed of 3 types of activities: an arrival or departure even (circle); a delay or queue event (a D symbol); and an action or service event (a square) (Figure 12). While these

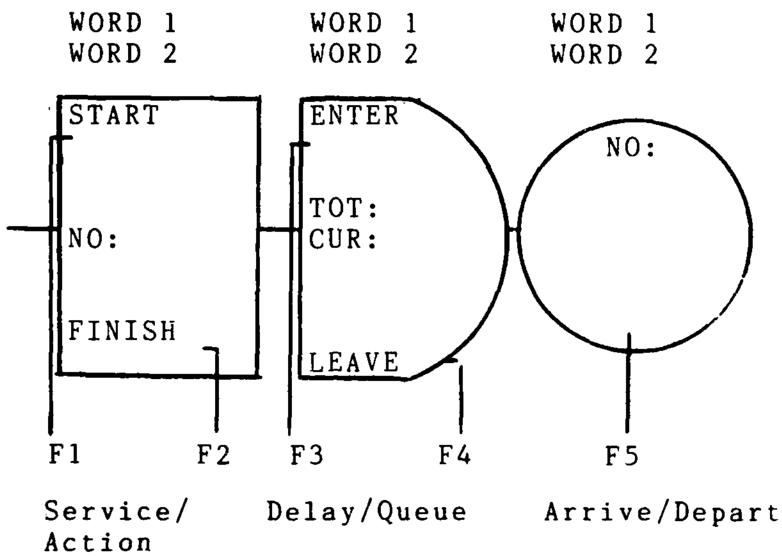

Figure 12. Activity/Event Symbols

symbols are geared to queueing systems, the action or service activity can represent any type of event or action; i.e., the time it takes to service a bank customer, or the time it takes to move a part from A to B, or the time it takes to perform a drilling operation, etc. [Caveat: This program is only seen as a starting point. The number or symbols and their layout/display can be changed and expanded in future program enhancements. The present layout was chosen to maximize the amount of data displayed on the schematic using normal text entry]

To build this plan, you start the program and at the main menu (Figure 13) select option 1, Develop Collection Diagram. The result is an input format (Figure 14). An

MAIN MENU

1. DEVELOP COLLECTION DIAGRAM

2. MODIFY COLLECTION DIAGRAM

3. COLLECT DATA

4. DATA FILE MAINTENANCE

5. EXIT PROGRAM

ENTER SELECTION:? [

Figure 13. Main Menu

explanation of the inputs follows:

A. Project NO/Ident (8 characters) - Used to uniquely identify the project. Can be either letters or numbers.

B. Collection Scenario NO (2 characters) - Used to identify the scenario. Can be either letters or numbers.

C. NO of Activities - Used to identify the number of collection activities for the particular scenario. The current program limits the maximum number of activities to 6.

For each activity the program requests the following:

D. Type of Activity (Select From Menu) - The menu is located above the input format (Figure 14). Can enter 1 of 3 type of activities as discussed above.

E. Parallel ACT(Y/N) - As discussed above, if activities of the same type as that defined in D are going

```
                    MENU - ACTIVITY TYPE

        1.  ARRIVAL/DEPARTURE ACTIVITY (ONE COLLECTION POINT)

        2.  DELAY/QUEUE/WAIT ACTIVITY (TWO COLLECTION POINTS)

        3.  SERVICE/ACTION/ACTIVITY (TWO COLLECTION POINTS)

        ********************************************

PROJECT NO/IDENT: ? A   *COLLECTION SCENARIO NO: B *NO OF ACTIVITIES:C *
             ACTIVITY          PARALLEL ACTIVITY NO:
    TYPE OF ACTIVITY (SELECT FROM MENU): D *  PARALLEL ACT(Y/N): E * NO F *
    DESCRIPTION OF ACTIVITY (UP TO 66 CHARACTERS):
   **                     G                                     **

    SHORT DESCRIPTION TO BE USED TO LABEL SCHEMATIC (ONE OR TWO WORDS - LIMIT
      TO 8 CHARACTERS):
           FIRST WORD: H     *SECOND WORD: I      **
```

Figure 14. Input Format To Build a Data Collection Schematic (Plan)

to be observed, you answer Y. This allows you to increase the number of activities you can observe and collect data on. If the answer is yes, enter Y. The cursor will go to F. If you enter N the cursor by-passes F and goes to G.

F. NO - The number of parallel activities associated with the current activity.

G. Description of Activity (66 characters) - Allows the user to enter a description of the activity/event. Should specify enough detail to sufficiently define the required data.

H. Short Description to be used to label schematic (one or two words, eight characters each) - On the collection plan schematic each activity diagram is labeled with these words. User should use key words to help in identifying the collection node. If there are parallel activities, a suggestion is to enter 1 of __ (like report page numbers) so you can identify which activity is being displayed.

The program will cycle through D-H for each activity. For parallel activities, the program will cycle G-H for each parallel activity/event. Thus each data collection point is individually defined and described.

When the last activity information is entered the program will display a schematic which the analyst will use to collect the data (Figure 15). The program asks the user if he/she wants to make any changes (not shown - currently limited to a change. Add a symbol and delete a symbol still have to be added to the program).

After all changes are made the program saves this information to a file. The program automatically develops a file name (Figure 16). The default disk drive for all data files is always drive B. The program then returns to the main menu.

At the main menu the user can develop another collection

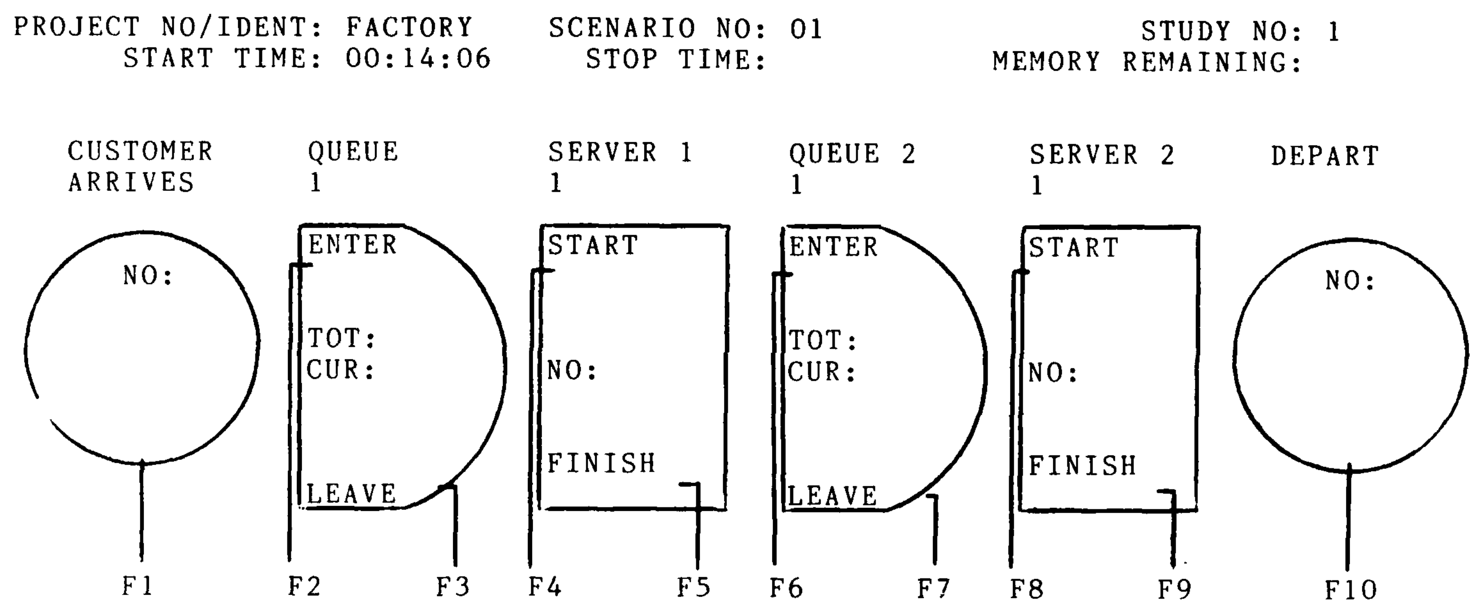

Figure 15. Data Collection Schematic (Plan)

```
*                                  **
 B: INPUT  P R __ __ __ __ __ __ . SDF

              First Four    Scenario
              Characters    Number
              Of Project
              Name
```

* Directory path not included in current program

** Simulation Definition File

Figure 16. Collection File Name

scenario, modify an existing one, or retrieve a scenario to collect data. [Option 4, File Maintenance is not completed]

Collecting Data

To collect data, option 3 is selected. The program then prompts the user for a project name, scenario number, and a study number (previously defined). The scenario collection plan is retrieved from the data disk. The program first displays a summary of the file information (Figure 17). Then the collection plan schematic is displayed (Figure 15).

When the plan was developed, an input key (function key) was automatically assigned and is displayed with the schematic. For arrival/departure nodes you collect one clock time each time the event occurs. This can be used to calculate interarrival times. The delay and activity nodes have two collection points; an enter or begin time and a leave or end time. The program then calculates a duration for the event. [Caveat: as currently written the duration or average delay time will only be accurate for FIFO, first-in-first-out,

PROJECT NO/IDENT: FACTORY   SCENARIO NO: 01   NO OF ACTIVITIES: 6

ACTIVITY #	TYPE ACTIVITY	PARALLEL ACTIVITY	SHORT ACTIVITY DESCRIPTION	F KEY(S)
1	1	1	CUSTOMER ARRIVES	F1
2	2	1	QUEUE 1	F2,F3
		2	QUEUE 1 2	
3	3	1	SERVER 1 1	F4,F5
		2	SERVER 1 2	
4	2	1	QUEUE 2 1	F6,F7
		2	QUEUE 2 2	
		3	QUEUE 2 3	
5	3	1	SERVER 2 1	F8,F9

PRESS ANY KEY TO CONTINUE

Figure 17. Data Collection Schematic (Plan) Summary

queue priority. Also, a time in system can only be determined for single-queue,single-server systems with FIFO priority. However, the collection scenario will not always define the entire system. The program was designed to collect data on various parts of a system and the order in which the nodes appear should reflect the order in which events are to be observed, not the order in which entities flow through a system. This does limit some of the validation data and needs to be corrected in future research.] Also calculated and archived are maximum number in queue and average number in queue.

To enter data for a particular event the appropriate function key is pressed. The program calculates the time, places the time data into an array, calculates the number in queue or in a service activity, and updates the display. If there are parallel activities, the computer prompts the user for the appropriate parallel activity number. This number is entered by pressing on of the numeral keys. The enter key does not have to be pressed for data entry. Thus, the maximum number of key inputs is two. The user does not have to enter any other identifying data. This was already done in the scenario development phase.

The other information displayed on the screen during data collection is the name of the project, the scenario number, and study number. When the collection routine is first called, a start time is calculated and displayed. The same thing occurs when data collection is terminated. The amount of memory remaining is also displayed. If the machine

has more than 256K of RAM, memory shouldn't be a major problem. The machine used to develop and run this program only had 256K which limits the number of collection entries per activity to between 50 and 75, and the number of parallel activities to 3.

To terminated data collection, the Ctr-F10 keys are pressed simultaneously. When this is done, the data is automatically stored to files on the data disk. Each activity has it's own file. Data Files are formatted as shown in Figure 19. Data stored includes the activity identifying data, the raw time data, the number of data entries, and for an arrival/departure event; interarrival times, maximum interarrival time and the average interarrival time. For the delay/queue activity, the raw enter/leave times are entered along with elapsed time, maximum number in queue, average number in queue, maximum delay time, and average delay time. For service/action activities, the raw enter/leave times are entered along with elapsed serviced times, maximum service times and average service times.

A file identifier is calculated as shown in Figure 19.

Programming Notes

It's necessary to point out that Advanced BASIC is required to run this program because of the schematic drawings. A complete program list is a Appendix C and includes a listing of the program variables and subroutines. The program makes extensive use of subroutines and should hopefully be

ARRIVAL/DEPARTURE
ACTIVITY

Project Name,Scenario No, Program Counter

Activity Type,Parallel Activity No,Program Counter

Long Activity Description

Short Activity Description 1,Short Activity Description 2

Study No,Date,Start Time,Terminate Time

Number of Observations in This File

Number of Entities or Actions Completed in this Activity,0

Observation Number, Event Time, Time Since Last Event (IAT)

.
.
.
.

Average Interarrival Time, Maximum Interarrival Time

DELAY OR SERVICE
ACTIVITY

First 7 entries the same as Arrival/Departure Event
.
.
Entry 8
Observation Number,Enter Time, Leave Time, Elapse Time
.
.
.
Average Elapse Time(Service Time), Maximum Elapse Time,
Average Queue Length (For Delay Activities)

Figure 18. Data File Contents Format

```
       *                                         **
B: INPUT __ __ __ __ __ __ __ __ __ . SCF

         First 2     |     STUDY    |
      CHARACTERS     |      NO
      Of PROJECT     |          ACTIVITY
      NAME           |          IDENTIFIER
                                NUMBER
                 SCENARIO
                   NO.
```

* Directory Path not included in current program.
** Simulation Collection File

Figure 19. Collection File Name

easy to follow. This section will briefly discuss a few of program techniques.

Accessing the Computer Clock. The program uses the same technique as MacMillan and Walker (1985). When the clock time is required, the time is entered into the character variable T$ using the TIME$ function (line number 5400). The character variable is then converted to a numerical value using the VAL function (line number 5755). The TIME$ function is in the format hr:min:sec. The numerical value is converted to minutes by multiplying the hours value by 60, adding it to the minutes value, and then adding this to the seconds value divided by 60. (for greater accuracy see MacMillan and Walker 1985).

Programming the Function Keys. The function keys are programmed using the INKEY$ function (subroutine 5300) and the extended ASCII codes. Because a function key returns a two

character set from the keyboard, the length of the key return is checked (lines 5370 and 5380). The INKEY$ value (FKEY$) is truncated and the ASCII value calculated using the ASC value. This value determines which F-key was pressed; F1-F10, Shift F1-F10, Ctl F1-F10, or Alt F1-F10. Once the program identifies which F-key was pressed it compares it to the input F-key assigned to each activity. (For more information see the BASIC User's Manual)

Schematic Symbols. The symbols are drawn using simple LINE and Circle functions (subroutines 1000,1200, and 1400). The position of the symbol is determined by the activity position number (1,2,3,4,5, or 6) and coordinates extracted from subroutine 2200.

Example

To demonstrate how a collection plan is developed and data collected, this section will examine a simple example.

The example consists of a model to simulate a small bank operation. The bank has 4 tellers with a single queue. Customers enter and exit through a single door. The bank layout is shown in Figure 20.

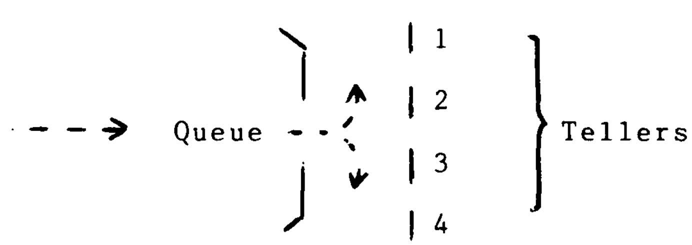

Figure 20. Bank Layout

To run the simulation, the interarrival times of the customers and the mean service time for each teller are required. To help validate the model, the average length of the queue and the average time spent in the queue are required.

To build the scenario, option 1 is selected at the main menu and the data plan input (see Figures 22-27). The schematic is then displayed and checked for accuracy (Figure 28). The program asks if there are any changes. There aren't, so N is entered and the program saves this information to file PRBANK01.SDF.

To collect data, option 3 is entered at the main menu and the program asks for a Project NO/IDENT. The name Bank is entered. It then asks for a scenario number. Number 1 is entered. Finally it asks for a study number. Since this is the first data collection study using scenario 1, the number 1 is entered (see Figure 29).

```
ENTER PROJECT NO/IDENT:? BANK
ENTER SCENARIO NO:? 1
ENTER STUDY #:? 1
```

Figure 29. Collection Plan Retrieval ID

The information defining the scenario is retrieved and the plan summary is displayed (Figure 30). The collection schematic is then displayed, Figure 28, and the collection

```
                    MENU - ACTIVITY TYPE

              1.  ARRIVAL/DEPARTURE ACTIVITY (ONE COLLECTION POINT)

              2.  DELAY/QUEUE/WAIT ACTIVITY (TWO COLLECTION POINTS)

              3.  SERVICE/ACTION/ACTIVITY (TWO COLLECTION POINTS)

              *****************************************************

PROJECT NO/IDENT: ? BANK    *COLLECTION SCENARIO NO: ? 1 *NO OF ACTIVITIES? 4*

                  ACTIVITY   1      PARALLEL ACTIVITY NO:  1

     TYPE OF ACTIVITY (SELECT FROM MENU):? 1*   PARALLEL ACT(Y/N):? N* NO:    *

     DESCRIPTION OF ACTIVITY (UP TO 66 CHARACTERS):
    *? CUSTOMER ENTERS BANK/PROCEEDS TO QUEUE                            **

     SHORT DESCRIPTION TO BE USED TO LABEL SCHEMATIC (ONE OR TWO WORDS - LIMIT
        TO 8 CHARACTERS):
             FIRST WORD:? CUST    *SECOND WORD:? ARRIVES[**
```

Figure 21. Input Display For Activity 1, Bank Example (Customer Arrives)

```
                    MENU - ACTIVITY TYPE

         1.  ARRIVAL/DEPARTURE ACTIVITY (ONE COLLECTION POINT)

         2.  DELAY/QUEUE/WAIT ACTIVITY (TWO COLLECTION POINTS)

         3.  SERVICE/ACTION/ACTIVITY (TWO COLLECTION POINTS)

         *****************************************************

PROJECT NO/IDENT: ? BANK    *COLLECTION SCENARIO NO: ? 1 *NO OF ACTIVITIES? 4*

               ACTIVITY   2      PARALLEL ACTIVITY NO:  1

     TYPE OF ACTIVITY (SELECT FROM MENU):? 2*   PARALLEL ACT(Y/N):? N* NO:   *

     DESCRIPTION OF ACTIVITY (UP TO 66 CHARACTERS):
    *? CUSTOMER WAITS FOR TELLER TO BE BREE/SINGLE QUEUE                **

     SHORT DESCRIPTION TO BE USED TO LABEL SCHEMATIC (ONE OR TWO WORDS - LIMIT
        TO 8 CHARACTERS):
              FIRST WORD:? CUST    *SECOND WORD:? WAITS[  **
```

Figure 22. Input Display For Activity 2, Bank Example (Customer Waits)

```
                    MENU - ACTIVITY TYPE

            1.  ARRIVAL/DEPARTURE ACTIVITY (ONE COLLECTION POINT)

            2.  DELAY/QUEUE/WAIT ACTIVITY (TWO COLLECTION POINTS)

            3.  SERVICE/ACTION/ACTIVITY (TWO COLLECTION POINTS)

            ****************************************************

PROJECT NO/IDENT: ? BANK    *COLLECTION SCENARIO NO: ? 1 *NO OF ACTIVITIES? 4*

                   ACTIVITY   3      PARALLEL ACTIVITY NO:  1

     TYPE OF ACTIVITY (SELECT FROM MENU):? 3*   PARALLEL ACT(Y/N):? Y* NO:? 4*

     DESCRIPTION OF ACTIVITY (UP TO 66 CHARACTERS):
    *? CUSTOMER SERVICED BY TELLER/1 OF 4 TELLERS                        **

     SHORT DESCRIPTION TO BE USED TO LABEL SCHEMATIC (ONE OR TWO WORDS - LIMIT
        TO 8 CHARACTERS):
              FIRST WORD:? TELLER1 *SECOND WORD:? 1 OF 4[ **
```

Figure 23. Input Display For Activity 3, Parallel Activity, 1, Bank Example (Teller 1 Service)

```
                MENU - ACTIVITY TYPE

        1.  ARRIVAL/DEPARTURE ACTIVITY (ONE COLLECTION POINT)

        2.  DELAY/QUEUE/WAIT ACTIVITY (TWO COLLECTION POINTS)

        3.  SERVICE/ACTION/ACTIVITY (TWO COLLECTION POINTS)

        ****************************************************

PROJECT NO/IDENT: ? BANK    *COLLECTION SCENARIO NO: ? 1 *NO OF ACTIVITIES? 4*

                 ACTIVITY   3      PARALLEL ACTIVITY NO:  2

     TYPE OF ACTIVITY (SELECT FROM MENU):? 3*   PARALLEL ACT(Y/N):? Y* NO:? 4*

     DESCRIPTION OF ACTIVITY (UP TO 66 CHARACTERS):
    *? CUSTOMER RECEIVES SERVICE BY TELLER 2/2 OF 4                      **

     SHORT DESCRIPTION TO BE USED TO LABEL SCHEMATIC (ONE OR TWO WORDS - LIMIT
        TO 8 CHARACTERS):
               FIRST WORD:? TELLER2 *SECOND WORD:? 2 OF 4[ **
```

Figure 24. Input Display For Activity 3, Parallel Activity 2, Bank Example (Teller 2 Service)

```
                MENU - ACTIVITY TYPE

        1.  ARRIVAL/DEPARTURE ACTIVITY (ONE COLLECTION POINT)

        2.  DELAY/QUEUE/WAIT ACTIVITY (TWO COLLECTION POINTS)

        3.  SERVICE/ACTION/ACTIVITY (TWO COLLECTION POINTS)

        *****************************************************

PROJECT NO/IDENT: ? BANK    *COLLECTION SCENARIO NO: ? 1 *NO OF ACTIVITIES? 4*

            ACTIVITY   3      PARALLEL ACTIVITY NO:  3

  TYPE OF ACTIVITY (SELECT FROM MENU):? 3*   PARALLEL ACT(Y/N):? Y* NO:? 4*

  DESCRIPTION OF ACTIVITY (UP TO 66 CHARACTERS):
 *? CUSTOMER RECEIVES SERVICE FROM TELLER 3/ 3 OF 4                   **

  SHORT DESCRIPTION TO BE USED TO LABEL SCHEMATIC (ONE OR TWO WORDS - LIMIT
     TO 8 CHARACTERS):
            FIRST WORD:? TELLER3 *SECOND WORD:? 3 OF 4[ **
```

Figure 25. Input Display For Activity 3, Parallel Activity 3, Bank Example (Teller 3 Service)

```
                    MENU - ACTIVITY TYPE

          1.  ARRIVAL/DEPARTURE ACTIVITY (ONE COLLECTION POINT)

          2.  DELAY/QUEUE/WAIT ACTIVITY (TWO COLLECTION POINTS)

          3.  SERVICE/ACTION/ACTIVITY (TWO COLLECTION POINTS)

          ****************************************************

PROJECT NO/IDENT: ? BANK    *COLLECTION SCENARIO NO: ? 1 *NO OF ACTIVITIES? 4*

              ACTIVITY   3      PARALLEL ACTIVITY NO:  4

    TYPE OF ACTIVITY (SELECT FROM MENU):? 3*   PARALLEL ACT(Y/N):? Y* NO:? 4*

    DESCRIPTION OF ACTIVITY (UP TO 66 CHARACTERS):
   *? CUSTOMER RECEIVES SERVICE FROM TELLER 4/ 4 OF 4                  **

    SHORT DESCRIPTION TO BE USED TO LABEL SCHEMATIC (ONE OR TWO WORDS - LIMIT
      TO 8 CHARACTERS):
           FIRST WORD:? TELLER4 *SECOND WORD:? 4 OF 4[ **
```

Figure 26. Input Display For Activity 3, Parallel Activity 4, Bank Example (Teller 4 Service)

```
                    MENU - ACTIVITY TYPE

          1.  ARRIVAL/DEPARTURE ACTIVITY (ONE COLLECTION POINT)

          2.  DELAY/QUEUE/WAIT ACTIVITY (TWO COLLECTION POINTS)

          3.  SERVICE/ACTION/ACTIVITY (TWO COLLECTION POINTS)

          ****************************************************

PROJECT NO/IDENT: ? BANK    *COLLECTION SCENARIO NO: ? 1 *NO OF ACTIVITIES? 4*

                ACTIVITY   4      PARALLEL ACTIVITY NO:  1

     TYPE OF ACTIVITY (SELECT FROM MENU):? 1*   PARALLEL ACT(Y/N):? N* NO:?  *

     DESCRIPTION OF ACTIVITY (UP TO 66 CHARACTERS):
    *? CUSTOMER DEPARTS BANK                                          **

     SHORT DESCRIPTION TO BE USED TO LABEL SCHEMATIC (ONE OR TWO WORDS - LIMIT
        TO 8 CHARACTERS):
              FIRST WORD:? CUST    *SECOND WORD:? DEPARTS[**
```

Figure 27. Input Display For Activity 4, Bank Example (Customer Departs Bank)

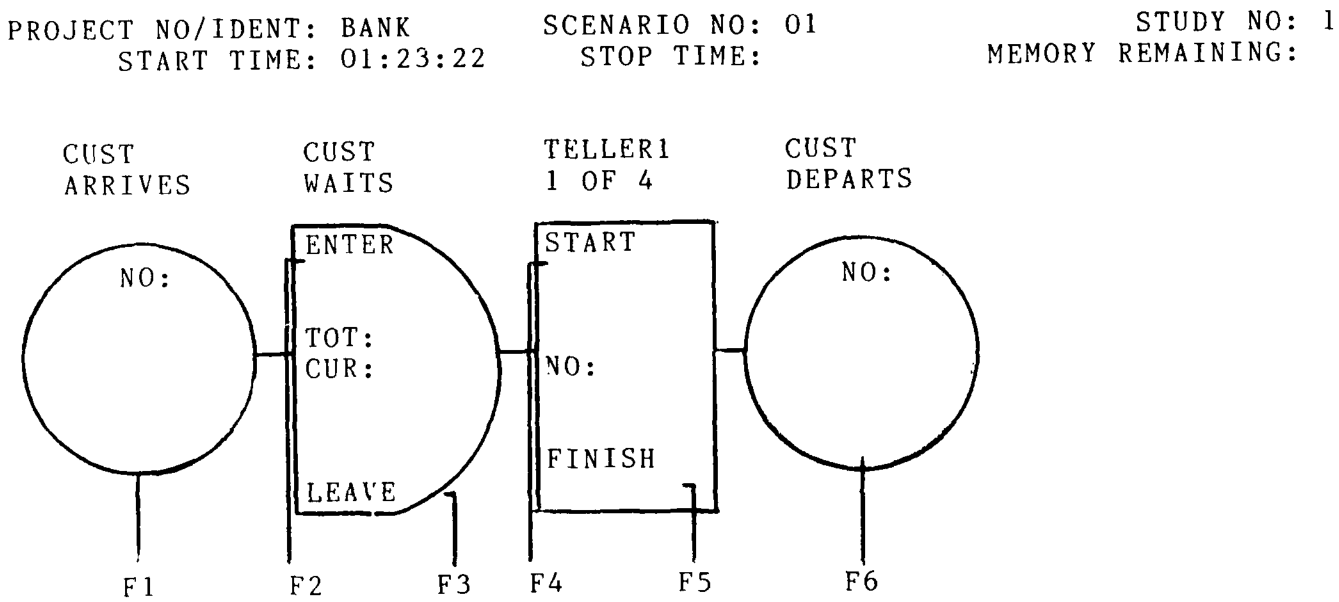

Figure 28. Data Collection Schematic (Plan) For Bank Example

PROJECT NO/IDENT: BANK   SCENARIO NO: 01   NO OF ACTIVITIES: 4

ACTIVITY #	TYPE ACTIVITY	PARALLEL ACTIVITY	SHORT ACTIVITY DESCRIPTION	F KEY(S)
1	1	1	CUST ARRIVES	F1
2	2	1	CUST WAITS	F2,F3
3	3	1	TELLER1 1 OF 4	F4,F5
		2	TELLER2 2 OF 4	
		3	TELLER3 3 OF 4	
		4	TELLER4 4 OF 4	
4	1	1	CUST DEPARTS	F6

PRESS ANY KEY TO CONTINUE

Figure 30. Summary of Data Collection Plan For Bank Example

process begun. Figure 31 shows a screen display during the collection process.

For this example the time was compressed and collection terminated after 12 arrivals. Ctl-F10 was pressed and the data was stored to files BA010101.SCF-BA010107.SCF. The file contents are shown in Figures 32-38 (refer to Figure 18 for format.

This is just a short example, but it does show the type of raw data that can be sent to the primary support system for analysis. It also highlights again the data that cannot currently be collected with this program, and some problems to watch for in the output data.

For example, because there are parallel service activities, you cannot determine the average length of time a customer spends in the bank, or even the average number of customers in the bank at any one time. You can approximate the length of time a customer spends in the bank by adding the average waiting time and average service time together. This is not very accurate, though, and this capability needs to be added.

Also, Figure 34, the data for teller 1, shows a negative average service time. This occurred because collection terminated with a customer still at teller number 1. This will often happen, so the data needs to be reviewed prior to analysis and this type of data removed.

Originally, the plan was to incorporate some statistical analysis software directly into the program. Instead, a

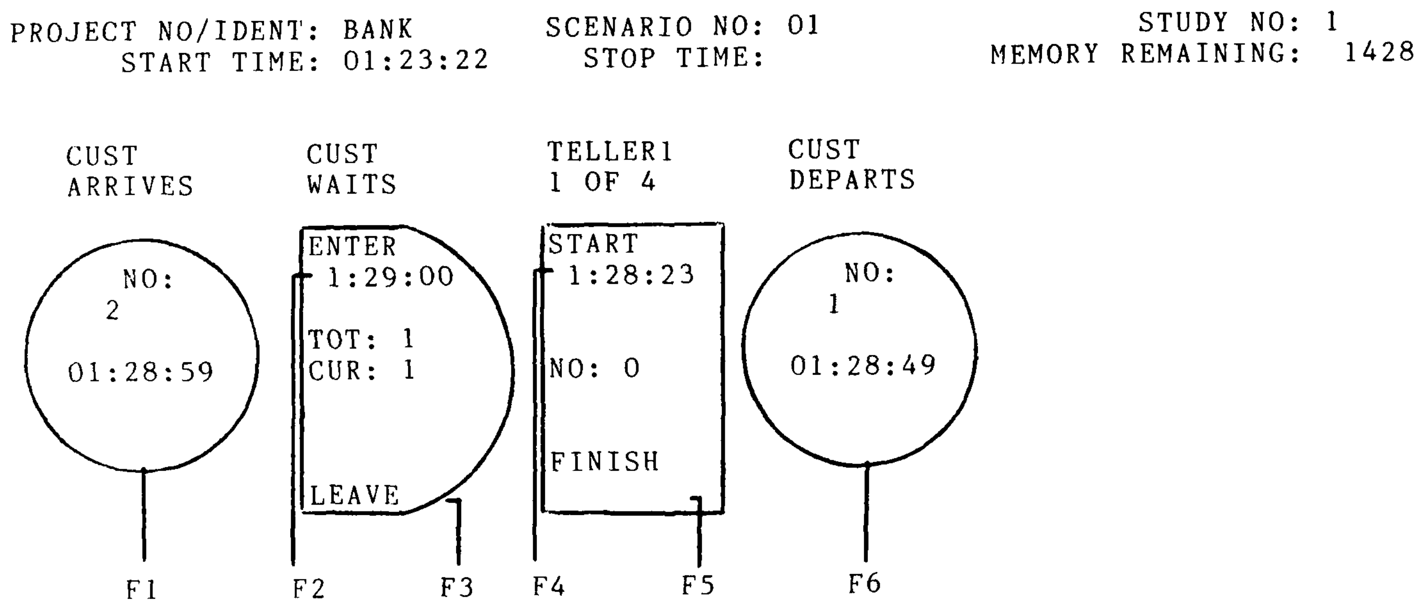

Figure 31. Display of Collection Schematic During Data Collection (Bank Example)

```
Type "exit" to return to the Application Selecto .
 B:\ B>TYPE BA010101.SCF
"BANK","01",4
1,1,1
"CUSTOMER ENTERS BANK/PROCEEDS TO QUEUE"
"CUST","ARRIVES"
"01","03-22-1987","22:43:06","22:58:03"
12
12,0
1,1368.117,5.016724
2,1368.367,.25
3,1368.567,.1999512
4,1368.583,1.672363E-02
5,1368.967,.3833008
6,1369.65,.6833496
7,1371.083,1.43335
8,1372.35,1.266602
9,1372.4,5.004883E-02
10,1373.467,1.06665
11,1374.717,1.25
12,1377.05,2.333374
1.073083,2.333374
```

Figure 32. Data File For Activity 1, Customer Arrivals (Bank Example)

```
Type "exit" to return to the Application Selector.
 B:\ B>TYPE BA010102.SCF
"BANK","01",4
2,1,2
"CUSTOMER WAITS FOR TELLER TO BE FREE/SINGLE QUEUE
"CUST","WAITS"
"01","03-22-1987","22:43:06","22:58:03"
8
8,1
1,1369,1370.45,1.449951
2,1369.683,1372.117,2.43335
3,1371.15,1373.9,2.75
4,1372.483,1374.183,1.700073
5,1372.5,1375.083,2.583374
6,1373.483,1375.483,2
7,1374.867,1376.183,1.31665
8,1377.1,1377.583,.4833985
1.6352,2.75,2.25
```

Figure 33. Data File For Activity 2, Customer Waits (Bank Example)

```
Type "exit" to return to the Application Selector.
 B:\ B>TYPE BA010103.SCF
"BANK","01",4
3,1,3
"CUSTOMER SERVICED BY TELLER/1 OF 4 TELLERS"
"TELLER1","1 OF 4"
"01","03-22-1987","22:43:06","22:58:03"
4
3,0
1,1368.2,1370.3,2.100098
2,1370.5,1374.933,4.43335
3,1375.133,1377.333,2.200073
4,1377.65,0,-1377.65
-273.7833,4.43335,0
```

Figure 34. Data File For Activity 3, Teller 1 (Bank Example)

```
Type "exit" to return to the Application Selector.
 B:\ B>TYPE BA010104.SCF
"BANK","01",4
3,2,3
"CUSTOMER RECEIVES SERVICE BY TELLER 2/ 2 OF 4"
"TELLER2","2 OF 4"
"01","03-22-1987","22:43:06","22:58:03"
2
2,0
1,1368.45,1374.017,5.566773
2,1374.367,1377.367,3
2.855591,5.566773,0
```

Figure 35. Data File For Activity 3, Teller 2 (Bank Example)

```
Type "exit" to return to the Application Selector.
 B:\ B>TYPE BA010105.SCF
"BANK","01",4
3,3,3
"CUSTOMER RECEIVES SERVICE FROM TELLER 3 / 3 OF 4"
"TELLER3","3 OF 4"
"01","03-22-1987","22:43:06","22:58:03"
3
3,0
1,1368.667,1371.95,3.283325
2,1372.217,1376.1,3.883301
3,1376.217,1377.383,1.166626
2.083313,3.883301,0
```

Figure 36. Data File For Activity 3, Teller 3 (Bank Example)

```
Type "exit" to return to the Application Selecto .
 B:\ B>TYPE BA010106.SCF
"BANK","01",4
3,4,3
"CUSTOMER RECEIVES SERVICE FROM TELLER 4 / 4 OF 4"
"TELLER4","4 OF 4"
"01","03-22-1987","22:43:06","22:58:03"
3
3,0
1,1368.733,1373.7,4.966675
2,1373.933,1375.3,1.366699
3,1375.517,1377.417,1.899902
2.058319,4.966675,0
```

Figure 37. Data File For Activity 3, Teller 4 (Bank Example)

```
Type "exit" to return to the Application Selecto .
 B:\ B>TYPE BA010107.SCF
"BANK","01",4
1,1,4
"CUSTOMER DEPARTS BANK"
"CUST","DEPARTS"
"01","03-22-1987","22:43:06","22:58:03"
11
11,0
1,1370.367,7.266724
2,1372.017,1.650024
3,1373.783,1.766602
4,1374.133,.3499756
5,1375,.8666992
6.1375.35,.3499756
7,1376.133,.7833252
8,1377.467,1.333374
9,1377.483,1.660156E-02
10,1377.5,1.672363E-02
11,1377.517,1.672363E-02
1.201396,1.766602
```

Figure 38. Data File For Activity 4, Customer Departs (Bank Example)

simple statistical package, such as Statistical Analysis from IIE Microsoftware is recommended. The input routines can be adjusted to read in data from these data files.

The program does organize and simplify the collection process. It's only a starting point, and many improvements need to be made.

CHAPTER 7

SUGGESTIONS FOR FUTURE RESEARCH

The research has shown that there has been a need for a more organized and structured approach for carrying out simulation projects. Furthermore, integrated simulation support systems, such as TESS, or concepts implied in a model management system, have sought to implement methodologies to answer this need. However, these systems still require quality data/information to operate and do not support the process of collecting the data. This thesis has, therefore proposed a system to support the data/information needs of these simulation support environments. The objective has been to develop a system concept for supporting the entire simulation process.

A system specification has been developed which seeks to achieve four general requirements in support of six functional areas. These requirements call for the proposed support system to help in:

1. organizing and guiding the collection effort;
2. organizing and archiving the data;
3. displaying the data for review, analysis, update and validation; and
4. producing required reports.

The six data collection functions include:

1. problem formulation
2. system description
3. model input data collection
4. project management
5. cost/benefit analysis
6. reports generation

To implement this specification this study proposed using a portable microcomputer as a collection workstation. With the ability to operate any of the software a regular PC can run, and its natural ability to archive data, it can provide the needed tools to support the collection effort. By developing standardized application programs for generic software, such as spreadsheets, data base managers, word processors, graphics software, and/or a common programming language such as BASIC, the analyst can adjust them to his/her own desires and to the particular needs of the project.

In order to demonstrate how this concept could be implemented a BASICA program was developed to support the model input data collection function. Despite this effort, the system is still at a highly conceptual stage and a great deal of work remains to be done. The remainder of the chapter will therefore outline some suggestions for future research, beginning with a summary of the work necessary to individually implement the six system functions and concluding with two general areas of research suggested by this study.

System Functions

Problem Formulation

As suggested, the methodology proposed by Balci and Nance (1985) appears to represent a very good starting point. Utilizing a programmable data base manager should provide the flexibility and logic necessary to guide the analyst through the process. This will require additional work in the following areas:

1. Development of standard questions/information needs formats and logic.
2. Development of data files to archive this data.
3. Identification and formats those standard files that will be used by the programs of the other system functions.

Input Data Collection

The program developed to support input data collection meets most of the specification. It's easy to use. File operation is invisible to the user, and a minimum number of keystrokes are required to enter data. The user only has to respond to input prompts to define a collection plan and a schematic collection format is produced automatically.

To fully meet specifications, however, it still needs to be coupled to a statistical software package so that some preliminary analysis can be conducted. As mentioned already, the IIE Statistical Analysis package would provide a good starting point. It is written in BASIC, so all that is

required is a subroutine to access and read the data from the files. An even simpler step in this direction, is a subroutine to summarize the statistics already calculated by the program.

Additionally, error trapping routines have to be developed for the data entry event. This will insure that an obviously bad key selection is not accepted until verified. If some bad data is collected, a routine is needed to cleanse the data prior to saving it to disk storage.

Also, as relates to specifications, the capability to capture more data for validation purposes is required. The ability to track entities through each collection activity will solve this problem, for the most part.

Finally, as it relates to input data collection, a program is needed to support data collection from documented sources. In some projects documents may represent the major source of data.

System Description

Here, future research should focus on developing a collection of software tools to help the analyst develop a static representations of the system. The various methods mentioned earlier, process charts, activity charts, flow diagrams, etc., are possible starting points. The flexibility of a small CAD package could represent one possibility. Also, the data collection program discussed in the previous chapter could provide a means to collect data for an activity

chart. The program that utilizes this data could then be developed to schematically display an activity chart.

Project Management & Cost/Benefit Analysis

Primary job here is to continue the task analysis in order to develop a good standard project schedule that can be used as a starting point to estimate and manage the project. A major issue is how the resources and cost to accomplish a project are to be estimated. Software development cost has always been difficult to estimate. To complicate this process, however, will be the use of the support systems. How will this affect the cost of the project. While they should hopefully make the process more efficient, it will result in additional computer time. So this aspect is currently an unknown and requires additional research.

Reports Generation

Here again standard formats have to be defined both for update reports and also project proposal formats.

General System Requirement

A key aspect is the design of the underlying data base structure. The concept calls for data that pertains to more than one functional area to be accessible to each of the applicable programs. This requires the identification and design of standard universal files. For example, for the project manager to track resources expended, each system

function will have to identify the name of the analyst and the time spent on the task. This information will have to be separated and placed in a unique file that is accessible by the project manager program. The goal is to have the data necessary to run each program in the central data base so it does not have to be reentered as the user moves from one program to another.

APPENDIX A

PROGRAM LISTING

TABLE 8

PROGRAM SUBROUTINES

Subroutine	Description
100	Develop Collection Schematic
200	Modify Collection Schematic
300	Collect Data
500	Display Main Menu
700	Draw Schematic
1000	Draw Activity Diagram (Square)
1200	Draw Arrival/Depart Diagram (Circle)
1400	Draw Delay Diagram
1600	Label Activity Diagram
1800	Label Arrival/Depart Diagram
2000	Label Delay Diagram
2200	Define Diagram Position
2400	Menu - Type Activity
2600	Define Collection Schematic (input format)
2800	Define Collection Schematic (Prompts for input)
3100	Clear Input Spaces
3300	Write Schematic Definition To File
3600	Retrieve Schematic Definition From File
3900	Summarize Project File (schematic definition)
5300	Collect Data
5500	Identify Which F-Key was Pressed
5700	Enter Data into Array
5820	Find Parallel Activity No.
6000	Update Display - Arrival/Depart
6300	Update Display - Delay
6500	Update Display - Service/Action
9000	Save Collection Data to File
9500	Save To File (Type 1 data - arrival/depart)
9700	Save to File (Type 2&3 data-delay and service)
10000	Print Diagram Header
11000	Modify Diagram
50000	Assign F-Key to Activity
51000	Clear Bottom Half of Screen

TABLE 9

KEY PROGRAM VARIABLES

Variable	Description
I	Activity number (1-6)
K	Parallel activity number (1-6)
J	Collection point on a Delay or service activity 1 = enter/begin 2 = leave/end
M	Observation number
ACTIVITYPOS(I)	Defines symbol position on the screen
ACTIVITYTYPE(I)	Defines type of symbol
LONGACTDESCRIPT(I,J)	Long description of activity
SHORTACTDESCRIPT(N,I,K)	Short activity description of activity
F$(N,I)	Function key label for diagram
F(N,I)	ASCII code for function key
F1(M,I,J,K)	Time event is logged
LAST(I,J,K)	Number of observations
NOTHROUGH(I,K)	Number of entities through an activity
NOIN(I,K)	Number of entities currently in a delay activity
MAXCONT(I,K)	Maximum number of entities in a delay activity
SUMNOIN(I,K)	Sum of the number of entities a delay activity
ETAVG	Average elapse time - delay or service activity

TABLE 9 -- CONTINUED

ETMAX	Maximum elapse time - delay or service activity
QLENAVG	Average number in a queue
ET	Elapse time (service or delay)
ETT	Total elapse time
SCENARIO$	Scenario number
PROJECT$	Project name
STUDY$	Study number
FIL$	File name
PARACTNO(I)	Number of parallel activities for activity I

```
5   CLS
10  CLS:SCREEN 2
20 DIM
ACTIVITYPOS(6).ACTIVITYTYPE(6),LONGACTDESCRIPT$(6,6),SHORTACT
DESCRIPT(2,6,6),F$(2,10),F(2,10),F1(75,6,2,4),LAST(6,2.6).NOT
HROUGH(6,6 ),NOIN(6,6),MAXCONT(6,6).SUMNOIN(6,6):IF G=1 THEN
RETURN
30 ON ERROR GOTO 60000
40  GOSUB 500              'DISPLAY MAIN MENU, GET SELECTION
50  ON IN GOSUB 100,200,300,70,70
60  GOTO 40
70  SYSTEM
90 '
95
'*************************************************************
*************** 96 '
97 '
100 '      SUBROUTINE 100 - DEVELOP COLLECTION SCHEMATIC
110 '
120 '
130  CLS:GOSUB 2400      'PRINT ACTIVITY MENU
135  GOSUB 2600          'PRINT INPUT FORMAT
140  GOSUB 2800          'GET INPUTS
145  CLS:GOSUB 700       'DRAW DIAGRAM
147  GOSUB 11000         'SEE IF USER WANTS TO MODIFY DIAGRAM
150  GOSUB 3300          'SAVE DIAGRAM TO FILE
155  RETURN
160  '
165
'*************************************************************
************** 170 '
175 '
200 '        SUBROUTINE 200 - MODIFY COLLECTION DIAGRAM
210 '
220 '
230  GOSUB 3600           'IDENTIFY WHICH FILE AND RETRIEVE
INFO 235  GOSUB 10000           'PRINT HEADER
240  GOSUB 700            'DRAW DIAGRAM
245  GOSUB 11000          'ASK WHAT IS TO BE CHANGED
250  GOSUB 3300           'SAVE TO FILE
255  RETURN
260 '
265
'*************************************************************
************** 270 '
275 '
300 '        SUBROUTINE 300 - COLLECT DATA
305 '
310 '
320  GOSUB 3600            'RETRIEVE DIAGRAM FROM FILE
325  GOSUB 10000           'PRINT HEADER INFO
330  GOSUB 700             'DRAW DIAGRAM
335  GOSUB 5300            'COLLECT DATA
```

```
340  GOSUB 9000            'END COLLECTION AND SAVE DATA TO
FILE 345  RETURN
350 '
355
'****************************************************************
************** 360 '
365 '
500 '        SUBROUTINE 500 - DISPLAY MAIN MENU
510 '
520  IF G= 1 THEN CLEAR:G=1:GOSUB 20
530  CLS
540  LOCATE 2,32:PRINT "DATA COLLECTION"
550  LOCATE 5,35:PRINT "MAIN MENU"
560  LOCATE 8,20:PRINT "1.  DEVELOP COLLECTION DIAGRAM"
570  LOCATE 10,20:PRINT "2.  MODIFY COLLECTION DIAGRAM"
580  LOCATE 12,20:PRINT "3.  COLLECT DATA"
590  LOCATE 14,20:PRINT "4.  DATA FILE MAINTENANCE (not
completed)" 600  LOCATE 16,20:PRINT "5.  EXIT PROGRAM"
620  LOCATE 22,15:INPUT "ENTER SELECTION:";IN
630  RETURN
640 '
650
'****************************************************************
************** 660 '
670 '
700 '        SUBROUTINE 700 - DRAW SCHEMATIC
710 '
720 '
730  FOR I = 1 TO N
740  J = ACTIVITYTYPE(I)
750  ON J GOSUB 1800,2000,1600
760  NEXT I
770  RETURN
780 '
785
'****************************************************************
************** 790 '
795 '
1000 '               SUBROUTINE 1000 - DRAW ACTIVITY DIAGRAM
1010 '
1020 '
1030 'KP=ACTIVITYPOS(I):GOSUB 2200
1040 LINE (X,54)-(X+70,128),,B
1050 LINE (X-18,91)-(X,91):LINE (X+70,91)-(X+88,91)
1060 LINE (X+2,66)-(X-3,66):LINE -(X-3,140)
1070 LINE (X+68,122)-(X+73,122):LINE - (X+73,140)
1080 RETURN
1090 '
1100
'****************************************************************
************ 1110 '
1120 '
```

```
1200 '        SUBROUTINE 1200 - DRAW ARRIVE/DEPART(CIRCLE)
DIAGRAM 1210 '
1220 'KP=ACTIVITYPOS(I):GOSUB 2200
1230 CIRCLE (X+35,91),35,,,,24/25
1240 LINE (X-18,91)-(X,91):LINE (X+35,126)-(X+35,140):LINE
(X+70,91)-(X+88,91) 1250 'GOSUB 1800
1260 RETURN
1270 '
1280
'************************************************************
************* 1290 '
1300 '
1400 '           SUBROUTINE 1400 - DRAW DELAY DIAGRAM
1410 '
1420 'KP=ACTIVITYPOS(I):GOSUB 2200
1430  PI=3.141593:CIRCLE (X+42,91),38,,3*PI/2,PI/2,24/25 1440
LINE (X+42,54)-(X,54):LINE -(X,128):LINE -(X+42,128) 1450
LINE (X+2,64)-(X-3,64):LINE -(X-3,140):LINE (X+72,118)-
(X+76,118):LINE -(X+76,140) 1460 LINE (X,91)-(X-18,91):LINE
(X+79,91)-(X+88,91)
1470 RETURN
1480 '
1490
'************************************************************
************* 1500 '
1510 '
1600 '        SUBROUTINE 1300 - LABEL ACTIVITY(SQUARE) DIAGRAM
1610 '
1620 KP=ACTIVITYPOS(I):GOSUB 2200
1630 LOCATE 5,NP1:PRINT SHORTACTDESCRIPT$(1,I,1)
1640 LOCATE 6,NP1:PRINT SHORTACTDESCRIPT$(2,I,1)
1650 LOCATE 8,NP1:PRINT "START"
1660 'LOCATF 9,NP1:PRINT TIME$
1670 LOCATE 12,NP1:PRINT "NO:":'LOCATE 12,NP1+3:PRINT 1111
1680 LOCATE 15,NP1:PRINT "FINISH"
1690 'LOCATE 16,NP1:PRINT TIME$
1700 LOCATE 19,NP1-1:PRINT F$(1,I):LOCATE 19,NP1+7:PRINT
F$(2,I) 1705 GOSUB 1000
1710 RETURN
1720 '
1730 '
1740
'************************************************************
************ 1750 '
1760 '
1800 '          SUBROUTINE 1600 - LABEL ARRIVE/DEPART(CIRCLE)
DIAGRAM 1810 '
1820 KP=ACTIVITYPOS(I):GOSUB 2200
1830 LOCATE 5,NP1:PRINT SHORTACTDESCRIPT$(1,I,1)
1840 LOCATE 6,NP1:PRINT SHORTACTDESCRIPT$(2,I,1)
1850  LOCATE 9,NP1+3:PRINT "NO:"
1860 ' LOCATE 10,NP1+1:PRINT 1111
1870 'LOCATE 12,NP1:PRINT TIME$
```

```
1880 LOCATE 19,NP1+3:PRINT F$(1,I)
1885 GOSUB 1200
1890 RETURN
1900 '
1910
'*****************************************************************
************* 1920 '
1930 '
2000 '            SUBROUTINE 2000 - LABEL DELAY DIAGRAM
2010 '
2020 KP=ACTIVITYPOS(I):GOSUB 2200
2030 LOCATE 5,NP1:PRINT SHORTACTDESCRIPT$(1,I,1)
2040 LOCATE 6,NP1:PRINT SHORTACTDESCRIPT$(2,I,1)
2050 LOCATE 8,NP1:PRINT "ENTER"
2060 'LOCATE 9,NP1:PRINT TIME$
2070 LOCATE 11,NP1:PRINT "TOT:":'LOCATE 11,NP1+4:PRINT 1111
2080 LOCATE 12,NP1:PRINT "CUR:":'LOCATE 12,NP1+4:PRINT 1111
2090 'LOCATE 15,NP1:PRINT TIME$
2100 LOCATE 16,NP1:PRINT "LEAVE"
2110 LOCATE 19,NP1-1:PRINT F$(1,I):LOCATE 19,NP1+7:PRINT
F$(2,I) 2115 GOSUB 1400
2120 RETURN
2130 '
2140
'*****************************************************************
************* 2150 '
2160 '
2200 '            SUBROUTINE 2200 - DEFINE DIAGRAM POSITION
2210 '
2220 '
2230 IF KP=1 THEN X=20:NP1=4
2240 IF KP=2 THEN X=126:NP1=17
2250 IF KP=3 THEN X=230:NP1=30
2260 IF KP=4 THEN X=334:NP1=43
2270 IF KP=5 THEN X=438:NP1=56
2280 IF KP=6 THEN X=542:NP1=69
2290 RETURN
2300 '
2310
'*****************************************************************
************** 2320 '
2330 '
2400 '        SUBROUTINE 2400 - MENU - ACTIVITY TYPE
2410 '
2420 '
2430  LOCATE 2,28:PRINT "MENU - ACTIVITY TYPE"
2440  LOCATE 4,15:PRINT "1.  ARRIVAL/DEPARTURE ACTIVITY (ONE
COLLECTION POINT)" 2450  LOCATE 6,15:PRINT "2.
DELAY/QUEUE/WAIT ACTIVITY (TWO COLLECTION POINTS)" 2460
LOCATE 8,15:PRINT "3.  SERVICE/ACTION/ACTIVITY (TWO
COLLECTION POINTS)" 2470  LOCATE 10,15:PRINT
"**************************************************" 2475
RETURN
```

```
2480  '
2490
'*************************************************************
************* 2500 '
2510 '
2600 '                SUBROUTINE 2600 - DEFINE COLLECTION
SCHEMATIC 2610 '
2620 FCOUNT=0
2630  LOCATE 12.1:PRINT "PROJECT NO/IDENT:"
2640  LOCATE 12,29:PRINT "*COLLECTION SCENARIO NO:"
2650  LOCATE 12,58:PRINT "*NO OF ACTIVITIES:":LOCATE
12,78:PRINT "*" 2660  LOCATE 14,21:PRINT "ACTIVITY"
2670  LOCATE 14,39:PRINT "PARALLEL ACTIVITY NO:"
2680  LOCATE 16,6:PRINT "TYPE OF ACTIVITY (SELECT FROM
MENU):":LOCATE 16,45:PRINT "*" 2690  LOCATE 16,49:PRINT
"PARALLEL ACT(Y/N):":LOCATE 16,70:PRINT "*" 2700  LOCATE
16,72:PRINT "NO:":LOCATE 16,78:PRINT "*"
2710  LOCATE 18,6:PRINT "DESCRIPTION OF ACTIVITY (UP TO 66
CHARACTERS):" 2720  LOCATE 19,5:PRINT "**":LOCATE 19,74:PRINT
"**"
2730  LOCATE 21,6:PRINT "SHORT DESCRIPTION TO BE USED TO
LABEL SCHEMATIC (ONE OR TWO WORDS - LIMIT" 2740  LOCATE
22,9:PRINT "TO 8 CHARACTERS):"
2750  LOCATE 23.16:PRINT "FIRST WORD:":LOCATE 23,37:PRINT
"*SECOND WORD:":LOCATE 23.60:PRINT "**" 2760  RETURN
2770 '
2780
'*************************************************************
************* 2790 '
2795 '
2800 '          SUBROUTINE 2800 - DEFINE COLLECTION SCHEMATIC -
INPUTS 2810  LOCATE 12,19:INPUT PROJECT$
2820  LOCATE 12,54:INPUT SCENARIO$
2825  IF LEN(SCENARIO$)=1 THEN SCENARIO$="0"+SCENARIO$
2830  LOCATE 12,75:INPUT N
2840  FOR I = 1 TO N
2850  ACTIVITYPOS(I)=I
2860  LOCATE 14,31:PRINT I
2870  LOCATE 16,42:INPUT ACTIVITYTYPE(I)
2880  LOCATE 16,67:INPUT AN$
2890  IF AN$="N" THEN GOTO 2910
2900  LOCATE 16,75:INPUT PARACTNO(I)
2910  IF PARACTNO(I)=0 THEN PARACTNO(I)=1
2920  FOR J= 1 TO PARACTNO(I)
2930  LOCATE 14,61:PRINT J
2940  LOCATE 19.6:INPUT LONGACTDESCRIPT$(I,J)
2950  LOCATE 23.27:INPUT SHORTACTDESCRIPT$(1,I,J)
2960  LOCATE 23,50:INPUT SHORTACTDESCRIPT$(2,I,J)
2970  IF PARACTNO(I)>1 THEN GOSUB 3150
2980  NEXT J
2985  IF CHG%=1 THEN CHG%=0:RETURN
2990  FCOUNT = FCOUNT + 1
3000  GOSUB 50000
```

```
3010  IF ACTIVITYTYPE(I)>1 THEN FCOUNT=FCOUNT+1
3020  GOSUB 3100
3030  NEXT I
3040  RETURN
3050  '
3060
'*************************************************************
************ 3070 '
3080 '
3100 '        SUBROUTINE 3100 - CLEAR INPUT SPACES
3110 '
3120 '
3130  LOCATE 14,62:PRINT " ":LOCATE 16,44:PRINT " ":LOCATE
16,69:PRINT " ":LOCATE 16,77:PRINT " " 3140  LOCATE
14,31:PRINT " "
3150 '     MINI-SUB - CLEAR ONLY FOR PARALLEL ACT INPUTS
3160  LOCATE 19,6:PRINT "
" 3170  LOCATE 23,27:PRINT "              "
3180  LOCATE 23,52:PRINT "          "
3190  RETURN
3200 '
3210 '
**************************************************************
************ 3220 '
3230 '
3300 '        SUBROUTINE 3300 - WRITE SCHEMATIC DEFINITION TO
FILE 3310 '
3320 '    ASSUMES DATA DISK IS IN DRIVE B
3330 '
3340 '
3350  IF LEN(SCENARIO$)=1 THEN SCENARIO$="0"+SCENARIO$
3360
FILE$="B:"+"PR"+LEFT$(PROJECT$,4)+RIGHT$(SCENARIO$,2)+".SDF"
3370 '
3380  OPEN "O",#1,FILE$
3390  WRITE #1,PROJECT$,SCENARIO$,N
3400  FOR I = 1 TO N
3410  WRITE #1,ACTIVITYPOS(I),ACTIVITYTYPE(I),PARACTNO(I)
3420  FOR J = 1 TO PARACTNO(I)
3430  WRITE #1,LONGACTDESCRIPT$(I,J)
3440  WRITE
#1,SHORTACTDESCRIPT$(1,I,J),SHORTACTDESCRIPT$(2,I.J) 3450
NEXT J
3460  WRITE #1,F$(1,I),F(1,I),F$(2,I),F(2,I)
3470  NEXT I
3480  CLOSE #1
3485  SCREEN 2
3486  RETURN
3490 '
3500 '
**************************************************************
*********** 3510 '
3520 '
```

```
3600 '          SUBROUTINE 3600 - RETRIEVE SCHEMATIC DEFINITION
FROM FILE 3610 '
3620 '
3630  CLS:SCREEN 1
3640  LOCATE 4,5:INPUT "ENTER PROJECT NO/IDENT:";PROJECT$
3650  LOCATE 6,5:INPUT "ENTER SCENARIO NO:";SCENARIO$
3655  IF LEN(SCENARIO$)=1 THEN SCENARIO$="0"+SCENARIO$
3656  LOCATE 8,5:INPUT "ENTER STUDY #:";STUDY$
3660  FILE$ =
"B:"+"PR"+LEFT$(PROJECT$,4)+RIGHT$(SCENARIO$,2)+".SDF" 3670 '
3680  OPEN "I",#1,FILE$
3690  INPUT #1,PROJECT$,SCENARIO$,N
3700  FOR I = 1 TO N
3710  INPUT #1,ACTIVITYPOS(I),ACTIVITYTYPE(I),PARACTNO(I)
3720  FOR J = 1 TO PARACTNO(I)
3730  INPUT #1,LONGACTDESCRIPT$(I,J)
3740  INPUT
#1,SHORTACTDESCRIPT$(1,I,J),SHORTACTDESCRIPT$(2,I,J) 3750
NEXT J
3760  INPUT #1,F$(1,I),F(1,I),F$(2,I),F(2,I)
3770  NEXT I
3780  CLOSE #1
3790  SCREEN 2
3795 '
3800 '
3810 '
************************************************************
*********** 3820 '
3830 '
3900 '           SUBROUTINE 3900 - DISPLAY PROJECT FILE
CONTENTS 3910 '
3920 '   LABEL PAGE
3930  LOCATE 1,8:PRINT "PROJECT NO/IDENT:":LOCATE 1,36:PRINT
"SCENARIO NO:":LOCATE 1,54:PRINT "NO OF ACTIVITIES:" 3940
LOCATE 3,4:PRINT "ACTIVITY#":LOCATE 3,17:PRINT "TYPE":LOCATE
3,27:PRINT "PARALLEL":LOCATE 3,39:PRINT "SHORT
ACTIVITY":LOCATE 3,57:PRINT "F KEY(S)":LOCATE 3,70:PRINT
"FILE"
3950  LOCATE 4,7:PRINT "#":LOCATE 4,15:PRINT
"ACTIVITY":LOCATE 4,27:PRINT "ACTIVITY":LOCATE 4,40:PRINT
"DESCRIPTION":LOCATE 4,68:PRINT "SEQUENCE #"
3960  LOCATE 5,4:PRINT "------------------------------------
----------------------------------" 3970 '    DISPLAY DATA
3980  LOCATE 1,26:PRINT PROJECT$:LOCATE 1,49:PRINT
SCENARIO$:LOCATE 1,72:PRINT N 3990  K=5:FC=1
4000  FOR I = 1 TO N
4010  K = K + 1
4020  LOCATE K,7:PRINT I:LOCATE K,18:PRINT ACTIVITYTYPE(I)
4030  K1=0
4040  FOR J = 1 TO PARACTNO(I)
4050  LOCATE K,30:PRINT J:LOCATE K,39:PRINT
SHORTACTDESCRIPT$(1,I,J)+" ";SHORTACTDESCRIPT$(2,I,J) 4060
IF K1 = 1 THEN GOTO 4085
```

```
4060  IF K1 = 1 THEN GOTO 4085
4070  IF ACTIVITYTYPE(I)=1 THEN K1=1:LOCATE K,58:PRINT
F$(1,I):GOTO 4085 4080  LOCATE K,58:PRINT
F$(1,I)+","+F$(2,I):K1=1
4085  LOCATE K,70:PRINT FC
4090  IF K=18 THEN GOSUB 5100
4100  K= K + 1:FC=FC+1
4110  NEXT J
4120  IF K=18 THEN GOSUB 5100
4130  NEXT I
4140  LOCATE 23,4:PRINT "PRESS ANY KEY TO CONTINUE"
4150  IF INKEY$="" THEN GOTO 4150
4160  RETURN
4170  '
4180
'*************************************************************
**************
4190 '
4200 '
5100  LOCATE 23,4:PRINT "PRESS ANY KEY TO CONTINUE"
5110  IF INKEY$="" THEN GOTO 5110
5120  GOSUB 51000
5130  K = 5
5140  RETURN
5150 '
5160
'*************************************************************
***************
5170 '
5180 '
5300 '          SUBROUTINE 5300 - COLLECT DATA
5310 '
5320 '
5330  FOR I1=1 TO 10
5340  KEY I1,""
5350  NEXT I1
5355  S$=TIME$:LOCATE 2,19:PRINT S$
5360  FKEY$=INKEY$:IF FKEY$="" THEN GOTO 5360
5370  IF LEN(FKEY$)=1 THEN GOTO 5360
5380  IF LEN(FKEY$)=2 THEN
FKEY$=RIGHT$(FKEY$,1):V=ASC(FKEY$) 5390  IF V = 103  THEN
E$=TIME$:LOCATE 2,43:PRINT E$:RETURN 5400  T$=TIME$
5410  GOSUB 5500      'IDENT F KEY
5420  GOSUB 5700      'ENTER INTO ARRAY
5430  ON ACTIVITYTYPE(I) GOSUB 6000,6300,6500
5440  GOTO 5360
5450 '
5460
'*************************************************************
**************
5470 '
5480 '
5500 '      SUBROUTINE 5500 - IDENTIFY WHICH FKEY WAS PRESSED
```

```
5510 '
5520 '
5530  FOR I = 1 TO N
5540  FOR J = 1 TO 2
5550  IF V = F(J,I) THEN GOTO 5580
5560  NEXT J
5570  NEXT I
5580  KP=I
5590  RETURN
5600 '
5610
'*********************************************** ******
**************
5620 '
5630 '
5700 '     SUBROUTINE 5700 - ENTER DATA COLLECTED INTO ARRAY
5710 '
5720 '
5730  IF PARACTNO(I)>1 THEN GOSUB 5820:GOTO 5750
5740  K = 1
5750  M = LAST(I,J,K):M=M+1
5755  T=(VAL(LEFT$(T$,2)))*60 + VAL(MID$(T$,4,2)) +
((VAL(RIGHT$(T$,2)))/60) 5756  LOCATE 2,72:PRINT FRE(0)
5760  F1(M,I,J,K)=T
5770  IF ACTIVITYTYPE(I)=1 THEN
NOTHROUGH(I,K)=NOTHROUGH(I,K)+1
5780  IF J=1 AND ACTIVITYTYPE(I)=2 THEN NOIN(I,K)=
NOIN(I,K)+1:IF NOIN(I,K)>MAXCON(I,K) THEN MAXCONT(I,K)=
NOIN(I,K)
5790  IF J=2 AND ACTIVITYTYPE(I)<>1 THEN
SUMNOIN(I,K)=SUMNOIN(I,K)+NOIN(I,K):NOIN(I,K)=NOIN(I,K)-
1:NOTHROUGH(I,K)=NOTHROUGH(I,K)+1:IF NOIN(I,K)<0 THEN
NOIN(I,K)=0
5800  LAST (I,J,K)=M
5810  RETURN
5815 '
5820 '      INTERNAL SUBROUTINE 5820 - FIND OUT PARALLEL
ACTIVITY NO
5830 '
5840  LOCATE 23,5:PRINT "ENTER PARALLEL ACTIVITY NUMBER"
5841  AN$=INKEY$:IF AN$="" THEN GOTO 5841
5842  IF AN$="1" THEN K=1
5843  IF AN$="2" THEN K=2
5844  IF AN$="3" THEN K=3
5845  IF AN$="4" THEN K=4
5846  IF AN$="5" THEN K=5
5847  IF AN$="6" THEN K=6
5850  LOCATE 23,5:PRINT "                                        "
5860  RETURN
5870 '
5880 '*********************************************
************************
5890 '
```

```
5895 '
6000 ' SUBROUTINE 6000 - UPDATE DISPLAY -ARRIVAL/DEPARTURE
6010 '
6020 '
6030  GOSUB 2200
6040  LOCATE 6,NP1:PRINT SHORTACTDESCRIPT$(2,I,K)
6050  LOCATE 10,NP1+1:PRINT NOTHROUGH(I,K)
6060  LOCATE 12,NP1:PRINT T$
6070  RETURN
6080 '
6090 '*****************************************
******************************
6100 '
6110 '
6300 '      SUBROUTINE 6300 - UPDATE DISPLAY - DELAY
6310 '
6320 '
6330  GOSUB 2200
6340  LOCATE 6,NP1:PRINT SHORTACTDESCRIPT$(2,I,K)
6350  IF J=2 THEN GOTO 6410
6360  LOCATE 9,NP1:PRINT T$
6370  LOCATE 11,NP1+4:PRINT NOTHROUGH(I,K)
6380  LOCATE 12,NP1+4:PRINT "    ":LOCATE 12,NP1+4:PRINT
NOIN(I,K)
6390  LOCATE 15,NP1:PRINT "        "
6400  GOSUB 1400:RETURN
6410  LOCATE 9,NP1:PRINT "        "
6420  LOCATE 11,NP1+4:PRINT NOTHROUGH(I,K)
6430  LOCATE 12,NP1+4:PRINT "    ":LOCATE 12,NP1+4:PRINT
NOIN(I,K):LOCATE 15,NP1:PRINT T$
6440  GOTO 6400
6450 '
6460 '************************************************
************************
6470 '
6480 '
6500 '   SUBROUTINE 6500 - UPDATE DISPLAY -SERVICE/ACTIVITY
6510 '
6520 '
6530  GOSUB 2200
6540  LOCATE 6,NP1:PRINT SHORTACTDESCRIPT$(2,I,K)
6550  IF J=2 THEN GOTO 6600
6560  LOCATE 9,NP1:PRINT T$
6570  LOCATE 12,NP1+3:PRINT NOTHROUGH(I,K)
6580  LOCATE 16,NP1:PRINT "        "
6590  GOSUB 1000:RETURN
6600  LOCATE 9,NP1:PRINT "        "
6610  LOCATE 12,NP1+3:PRINT NOTHROUGH(I,K)
6620  LOCATE 16,NP1:PRINT T$
6630  GOTO 6590
6640 '
6650 '**********************************************
*************************
```

```
6660 '
6670 '
9000 '        SUBROUTINE 9000 - SAVE COLLECTION DATA TO FILES
9010 '
9020 '
9030  FOR I = 1 TO N
9040  FOR K = 1 TO PARACTNO(I)
9050  COUNT = COUNT + 1:COUNT$=STR$(COUNT)
9055  IF COUNT<10 THEN K1=1 ELSE K1=2
9056  COUNT$=RIGHT$(COUNT$,K1)
9060  IF LEN(COUNT$)=1 THEN COUNT$="0"+COUNT$
9065  IF I>1 THEN GOTO 9156
9150  IF LEN(SCENARIO$)=1 THEN SCENARIO$="0"+SCENARIO$
9153  IF LEN(STUDY$)=1 THEN STUDY$="0"+STUDY$
9155  FIN$="B:"+LEFT$(PROJECT$,2)+RIGHT$(SCENARIO$,2)+STUDY$
9156  FIL$=FIN$+RIGHT$(COUNT$,2)+".SCF"
9170  OPEN "O",#1,FIL$
9180  WRITE #1,PROJECT$,SCENARIO$,N
9190  WRITE #1,ACTIVITYTYPE(I),K,I
9200  WRITE #1,LONGACTDESCRIPT$(I,K)
9210  WRITE #1,SHORTACTDESCRIPT$(1,I,K),SHORTACTDESCRIPT$
(2,I,K)
9220 WRITE #1,STUDY$,DATE$,S$,E$
9230  WRITE #1,LAST(I,1,K)
9240  WRITE #1,NOTHROUGH(I,K),MAXCONT(I,K)
9245  IF ACTIVITYTYPE(I)=2 THEN
QLENAVG=SUMNOIN(I,K)/LAST(I,1,K) ELSE QLENAVG=0 9250
IF ACTIVITYTYPE(I)=1 THEN GOSUB 9500
9260  IF ACTIVITYTYPE(I)<>1 THEN GOSUB 9700
9270  CLOSE #1
9280  NEXT K
9290  NEXT I
9300  RETURN
9310 '
9320
'*************************************************************
**************
9330 '
9340 '
9500 '        SUBROUTINE 9500 - SAVE TO FILE TYPE 1 DATA
9510 '
9520 '
9530  ET=0:ETT=0:ETMAX=0:ETAVG=0
9540
ST=(VAL(LEFT$(S$,2)))*60+(VAL(MID$(S$,4,2)))+(VAL(RIGHT$(S$,
2)))/60
9550  ET=F1(1,I,1,K)-ST:M=1
9560  WRITE #1,M,F1(1,I,1,K),ET
9570  ETT=ETT+ET
9580  FOR M = 2 TO LAST(I,1,K)
9590  ET=F1(M,I,1,K)-F1(M-1,I,1,K)
9600  WRITE #1,M,F1(M,I,1,K),ET
9610  ETT=ETT+ET
```

```
9620  IF ET>ETMAX THEN ETMAX=ET
9630  NEXT M
9640  ETAVG=ETT/M
9650  WRITE #1,ETAVG,ETMAX
9660  RETURN
9670 '
9680
'**************************************************************
**************
9690 '
9695 '
9700 '          SUBROUTINE 9700 - SAVE TO FILE TYPE 2&3 DATA
9710 '
9720 '
9730  ET=0:ETT=0:ETMAX=0:ETAVG=0
9740  FOR M = 1 TO LAST(I,1,K)
9750  ET=F1(M,I,2,K)-F1(M,I,1,K)
9760  WRITE #1,M,F1(M,I,1,K),F1(M,I,2,K),ET
9770  ETT=ETT+ET
9780  IF ET>ETMAX THEN ETMAX=ET
9790  NEXT M
9800  ETAVG=ETT/M
9810  WRITE #1,ETAVG,ETMAX,QLENAVG
9820  RETURN
9830 '
9840
'**************************************************************
**************
9850 '
9860 '
10000 '       SUBROUTINE 10000 - PRINT DIAGRAM HEADER
10010 '
10020 '
10030  CLS
10040  LOCATE 1,1:PRINT "PROJECT NO/IDENT: ";PROJECT$
10050  LOCATE 1,30:PRINT "SCENARIO NO: ";SCENARIO$
10060  LOCATE 1,62:PRINT "STUDY NO: ";STUDY$
10070  LOCATE 2,7:PRINT "START TIME:"
10080  LOCATE 2,32:PRINT "STOP TIME:"
10090  LOCATE 2,54:PRINT "MEMORY REMAINING:"
10100  RETURN
10110 '
10120
'**************************************************************
*************
10130 '
10140 '
11000 '          SUBROUTINE 11000 - MODIFY DIAGRAM
11005 '
11006 '
11010  FCOUNT=0
11015  LOCATE 22,1:INPUT "DO YOU WANT TO MAKE ANY CHANGES
(Y/N):";CH$
```

```
11020  IF CH$="N" THEN GOTO 11125
11025  GOSUB
11500:LOCATE 22,1:INPUT "DO YOU WANT TO CHANGE THE SCENARIO
NO (Y/N):";AN$
11030  IF AN$="N" THEN GOTO 11040
11035  GOSUB 11500:LOCATE 22,1:INPUT "INPUT NEW SCENARIO
NO.:";SCENARIO$
11040  GOSUB
11500:LOCATE 22,1:PRINT "Do you want to CHANGE
a symbol (1), ADD a symbol (2), or DELETE a symbol (3)."
11045  LOCATE 23,1:INPUT "ENTER 1,2,OR 3:";CH
11050  ON CHGOTO 11055,11135,11135
11055  GOSUB 11500:LOCATE 22,1:INPUT "WHICH ACTIVITY TO YOU
WANT TO CHANGE (1,2,3,4,5,OR 6):";I:CLS
11060  GOSUB 2400              'PRINT ACTIVITY MENU
11065  GOSUB 2600                    'PRINT INPUT FORMAT
11070  CHG%=1:GOSUB 2850
11075  FOR I = 1 TO N              'RELABEL FKEY LABELS
11080  ACTIVITYPOS(I)=I
11085  FCOUNT=FCOUNT+1
11090  GOSUB 50000
11095  IF ACTIVITYTYPE(I)=2 OR ACTIVITYTYPE(I)=3 THEN
FCOUNT=FCOUNT+1 11100  NEXT I
11105  GOSUB 10000
11110  GOSUB 700
11115  LOCATE 22,1:INPUT "DO YOU WANT TO MAKE ANOTHER CHANGE
(Y/N):";CH$ 11120  IF CH$="Y" THEN FCOUNT=0:GOTO 11040
11125  RETURN
11130 '          ADD A SYMBOL
11135  GOSUB 11500:LOCATE 22,1:PRINT "PROGRAMMING IN
PROGRESS":FOR Y=1 TO 1000:Y=Y+1:NEXT Y
11140  GOTO 11115
11500 '          SUBROUTINE 11500 - CLEAR LINE 22
11505 '
11510 '
11515  LOCATE 22,1:PRINT "
"
11516  LOCATE 23.1:PRINT "
"
11520  RETURN
11525 '
11530
'*****************************************************************
*************
11535 '
11540 '
50000 '       SUBROUTINE 50000 - ASSIGN F KEY
50010 '
50020 '
50030  IF FCOUNT=1 THEN
F$(1,I)="F1":F(1.I)=59:F$(2,I)="F2":F(2,I)=60:GOTO 50150
50040  IF FCOUNT=2 THEN
F$(1.I)="F2":F(1,I)=60:F$(2,I)="F3":F(2,I)=61:GOTO 50150
```

```
50050  IF FCOUNT=3 THEN
F$(1,I)="F3":F(1,I)=61:F$(2,I)="F4":F(2,I)=62:GOTO 50150
50060  IF FCOUNT=4 THEN
F$(1,I)="F4":F(1,I)=62:F$(2,I)="F5":F(2,I)=63:GOTO 50150
50070  IF FCOUNT=5 THEN
F$(1,I)="F5":F(1,I)=63:F$(2,I)="F6":F(2,I)=64:GOTO 50150
50080  IF FCOUNT=6 THEN
F$(1,I)="F6":F(1,I)=64:F$(2,I)="F7":F(2,I)=65:GOTO 50150
50090  IF FCOUNT=7 THEN
F$(1,I)="F7":F(1,I)=65:F$(2,I)="F8":F(2,I)=66:GOTO 50150
50100  IF FCOUNT=8 THEN
F$(1,I)="F8":F(1,I)=66:F$(2,I)="F9":F(2,I)=67:GOTO 50150
50110  IF FCOUNT=9 THEN
F$(1,I)="F9":F(1,I)=67:F$(2,I)="F10":F(2,I)=68:GOTO 50150
50120  IF FCOUNT=10 THEN F$(1,I)="F10":F(1,I)=68:F$(2,I)="S-
F1":F(2,I)=84:GOTO 50150 50130  IF FCOUNT=11 THEN
F$(1,I)="S-F1":F(1,I`=84:F$(2,I)="S-F2":F(2,I)=85:GOTO 50150
50140  IF FCOUNT=12 THEN F$(1,I)="S=F2":F(1,I)-
85:F$(2,I)="S-F3":F(2,I)=86:GOTO 50150 50150  IF
ACTIVITYTYPE(I)=1 THEN F(2.I)=0
50160  RETURN
50170 '
50180
'************************************************************
*************
50190 '
50200 '
51000 '    SUBROUTINE 51000 - CLEAR BOTTOM HALF OF SCREEN
FOR DATA DISPLAY
51010 '
51020  FOR K = 6 TO 18
51030  LOCATE K,3:PRINT "
"
51040  NEXT K
51050  RETURN
51060 '
51070
'************************************************************
**************
51080 '
51090 '
60000 IF ERR=16 THEN E=E+1:RESUME 0
```

LIST OF REFERENCES

Annino, Joseph S., and Russell, Edward C., "The Ten Most Frequent Causes of Simulation Analysis Failure - and how to avoid them." Simulation, June 1979, pp. 137-141.

______, BASIC (users manual), IBM, 1981.

Balci, Osman, and Nance, Richard E., "Formulated Problem Verification As An Explicit Requirement of Model Credibility." Simulation, August 1985, pp. 76-86.

Banks, Jerry, and Carson, John S., Discrete-Event System Simulation. Englewood Cliffs: Printice-Hall, INC., 1984.

Barnes, Ralph M., Motion and Time Study, Design and Measurement of Work (6th ed.). New York: John Wiley & Sons, INC., 1968.

Bloom, Michael, "Personal Computers Extend The Reach of Data Acquisition Systems." Computer Design, April 15, 1986, pp. 43-60.

______, "Catalog of Simulation Software." Simulation, October 1985, pp. 196-205.

Cobbin, Phillip, "A Tutorial On The SIMPLE_1 Simulation Environment", Sales literature received from Sierra Simulations and Software, Campbell, CA., 1987.

Collella, A.M., Carlino, D.J., and O'Sullivan, M.J., System Simulation Methods and Applications. Lexington: D.C. Heath and Company, 1974.

Davis, Richard M., Thesis Projects in Science and Engineering, A Complete Guide From Problem Selection to Final Presentation. New York: St Martin's Press, 1980.

Dossett, Royal J. "Using Data Collectors With The IBM-BC For Work Measurement." Computers & Industrial Engineering 11 (1986): 476-480.

Evers, Kenneth H.; Bachert, Robert F.; and Santucci, Patrick R. "SADT/SAINT: Large Scale Analysis Simulation Methodology", 1981 Winter Simulation Conference Proceedings. 185-191 1981.

Golden, Donald G. "Software Engineering Considerations for the Design of Simulation Languages." Simulation, October 1985, pp. 169-178.

Gordon, G. System Simulation. Englewood Cliffs: Printice Hall Inc., 1978.

Grant, John W., and Weiner, Steven A. "Factors to Consider in Choosing a Graphically Animated Simulation System." Industrial Engineering, August 1986, p. 37.

Gray, Paul. "Managing Simulation Projects", Proceedings of the 1978 Winter Simulation Conference. pp. 63-68, 1978.

Healy, Kevin J. "Cinema Tutorial", Proceedings of the 1985 Winter Simulation Conference. pp. 94-97, 1985.

Henricksen, James O. "Discrete Event Simulation Languages: Current Status and Future Directions". Proceedings of the 1984 Winter Simulation Conference. pp. 83-88, 1984.

Hollocks, Brian. "The Impact of the Microcomputer in Simulation" Operational Research '84. Proceedings of the Tenth International Conference, 1984.

Huhn, Ronald M., and Comer, Edward R. "Applications of Modern Software Techniques to Modeling and Simulation." 1981 Winter Simulation Conference Proceedings. pp. 101-112, 1981.

Jones, Edward. Using dBASE III. Berkeley: McGraw-Hill, 1985.

Kelton, W. David. "Input Data Collection and Analysis." Proceedings of the 1984 Winter Simulation Conference. pp. 91-95, 1984.

Kerckhoffs, Eugene J., and Vansteenkiste, Ghislain C. "The Impact of Advanced Information Processing on Simulation -An Illustrative Review." Simulation, January 1986, pp. 17-26.

Kerzner, Harold. Project Management, A Systems Approach to Planning, Scheduling, and Controlling. New York: Van Nostrand Reinhold Company, 1984.

Ketcham, Michael G.; Hogg, Gary L.; and Shannon, Robert E. "Memory Management, Data Structures, and the Development of Advanced Simulation Software in a Microcomputer Environment." Modeling and Simulation on Microcomputers: 1985, Society for Computer Simulation, pp. 67-70, 1985.

Kisko, Thomas. "Using a Pocket Computer as a Time Study Data Collection Tool", Proceedings of the 8th Annual Conference on Computers and Industrial Engineering. Orlando, Fl: pp. 485-489, 1986.

Law, Averill and Kelton, W. David, Simulation Modeling and Analysis. New York: McGraw-Hill Book Co., 1982.

MacMillan, Jeff S. and Walker, Arlen P. "Portable Microcomputers Function as Data Controllers." Industrial Engineering, January 1985, pp. 18-26.

Martin, F.F., Computer Modeling and Simulation. New York: John Wiley & Sons, INC., 1968.

Martin, John C. "Program, Portable Microprocessor Allow Direct On-Site Input of MTM-1 Data." Computerized Work Measurement. Norcross: Industrial Engineering and Management Press, 1984.

McDermott, Kevin J. "Microcomputer and Spreadsheep Software Make Time Studies Less Tedious, More Accurate." Computerized Work Measurement, Norcross: Industrial Engineering and Management Press, 1984.

Miller, Annita and Pare, John. "Managing Simulation Projects." 1986 Annual Simulation Symposium, pp. 43-57, 1986.

Nance, Richard E."A Tutorial View of Simulation Model Development." Proceedings of the 1983 Winter Simulation Conference. pp. 16-20, 1983.

Nance, Richard E. "Chairman's Corner - When You Are Hot and When You Are Not." Simuletter, April 1985, pp. 2-3.

Nance, Richard E.; Mezaache, Ahmed L.; and Overstreet, C.Michael. "Simulation Model Management: Resolving the Technology Gaps." 1981 Winter Simulation Conference Proceedings. pp. 173-179, 1981.

Norgaard, Kritian. "Microcomputer Usage on the Production Floor", Computers & Industrial Engineering, 9 (1985) 405-417.

O'Keefe, Robert. "Simulation and Expert Systems - A Taxonomy and Some Examples." Simulation, January 1986, pp. 10-16.

Oren, Tuncer I., and Zeigler, Bernard P. "Concept for Advanced Simulation Methodologies." Simulation, March 1979, pp. 69-81.

Pritsker, A.A., Introduction to Simulation and SLAM. New York: John Wiley & Sons, 1986.

Redman, S., User Friendly Guide to Lap Portables, New York: McGraw-Hill, 1985.

Savage, Dennis F. and Keevan, Richard J. "Automating Your Current Work Measurement Data on Spreadsheet Software." Computerized Work Measurement. Norcross: Industrial Engineering and Management Press, 1984.

Schmidt, J.W. "Introduction to System Analysis, Modeling and Simulation." Proceedings of the 1985 Simulation Conference, pp 3-13, 1985.

Schruben, Lee. "Modeling Systems Using Discrete-Event Simulation." Proceedings of the 1983 Winter Simulation Conference, pp 101-107, 1983.

Shannon, Richard E. System Simulation: the art and science. New Jersey: Printice-Hall, 1975.

Shannon, Richard E. "Keynote Address - Artificial Intellegence and Simulation." Proceedings of the 1984 Winter Simulation Conference, pp. 3-9, 1984.

Sprague, David A. and Scholton, Mark. "Microcomputer Assisted Standard Data Development." Computerized Work Measurement. Norcross: Industrial Engineering and Management Press, 1984.

Standridge, Charles R. "Using the Simulation Data Language (SDL)." Simulation, September 1981, pp. 73-81.

Standridge, Charles R. "The Simulation Data Language (SDL): Applications and Examples." Simulation, October 1981, pp. 119-130.

Standridge, Charles R. "Performing Simulation Projects With The Extended Simulation System." Simulation, December 1985, pp. 283-291.

Standridge, Charles R. and Walker, Steven A. "An Introduction to Simulation Support Software." Proceedings of the 1983 Winter Simulation Conference. pp. 381-384 1984.

Standridge, Charles R. "Animating Simulation Using TESS." Computers & Industrial Engineering, 10 (1986), pp. 121-134.

Standridge, Charles R.; Vaughan, David K.; and Sale, Mary L. "A Tutorial on TESS: The Extended Simulation System." Proceedings of 1985 Winter Simulation Conference, pp. 73-79, 1985.

Waite, Mitchell and Morgan, Christopher L. Graphics Primer for the IBM PC, Berkeley: McGraw-Hill, 1983.

Weinberg, Victor. Structured Analysis, New York: Youdon Press, 1979.

Wilkerson, David L. "Computer-Aided Standard Data Development For Work Measurement." Computerized Work Measurement, Norcross: Industrial Engineering and Management Press, pp 52-59, 1984.

Womack, John W. and Marr, George R. "Desktop Data Bases For Simulation Development and Documentation." Modeling and Simulation On Microcomputers, Society for Computer Simulation, pp. 63-66, 1985.

Wygant, Robert M. "Data File Management." Proceedings of The 8th Annual Conference on Computers and Industrial Engineering, pp 367-371, 1986.

Zeigler, Bernard P. "System-Theoretic Representation of Simulation Models." IIE Transactions, March 1984, pp. 19-34.

Zeigler, M.S.; Elzas, G.J.; Klir, G.J.; and Oren, T.I. eds. Methodology in Systems Modeling and Simulation, North Holland Publishing Company, 1979.